Titre: Title:	Computer Vision Tools for Rodent Monitoring
Auteur: Author:	Rana Farah
Date:	2013
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Farah, R. (2013). Computer Vision Tools for Rodent Monitoring [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/1198/

UNIVERSITÉ DE MONTRÉAL

COMPUTER VISION TOOLS FOR RODENT MONITORING

RANA FARAH

DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION

DU DIPLÔME DE PHILOSOPHIÆ DOCTOR

(GÉNIE INFORMATIQUE)

JUILLET 2013

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

COMPUTER VISION TOOLS FOR RODENT MONITORING

présentée par : FARAH Rana

en vue de l'obtention du diplôme de : Philosophiæ Doctor

a été dûment acceptée par le jury d'examen constitué de :

M. PAL Christopher J., Ph.D., président

M. LANGLOIS J.M. Pierre, Ph.D., membre et directeur de recherche

M. BILODEAU Guillaume-Alexandre, Ph.D., membre et codirecteur de recherche

M. KADOURY Samuel, Ph.D., membre

Mme LAPORTE Catherine, Ph.D., membre

For my angels, Mom, Dad and Nizos

ACKNOWLEDGEMENT

I would like to express my sincere and greatest gratitude to my supervisors, Dr. Langlois and Dr. Bilodeau. I am thankful for their constant support and encouragement, for their constant availability and constructive advice and comments.

I am also thankful for my colleagues in the LASNEP and LITIV, for the good times, mutual support and encouragement that we shared. I would also like to thank Mr. Pier-Luc St-Onge for his technical support that helped advance my work. I would like to extend my gratitude to Dr. Sébastien Desgent and Mrs Sandra Duss for their help in providing and manipulating the rodent used in our experiments. Their input was crucial for our experiments. I will always be thankful to the faculty, staff and students of the department for they helped me expand my knowledge and expertise and appreciate my work even more. I would also like to thank them for every bit of help they provided each in his way.

My eternal gratitude would be for my family and friends for their unconditional encouragement and support.

Finally, I would like to thank the members of the jury for their valuable comments on my thesis.

ABSTRACT

Rodents are widely used in biomedical experiments and research. This is due to the similar characteristics that they share with humans, to the low cost and ease of their maintenance and to the shortness of their life cycle, among other reasons.

Research on rodents usually involves long periods of monitoring and tracking. When done manually, these tasks are very tedious and prone to error. They involve a technician annotating the location or the behavior of the rodent at each time step. Automatic tracking and monitoring solutions decrease the amount of manual labor and allow for longer monitoring periods. Several solutions have been provided for automatic animal monitoring that use mechanical sensors. Even though these solutions have been successful in their intended tasks, video cameras are still indispensable for later validation. For this reason, it is logical to use computer vision as a means to monitor and track rodents.

In this thesis, we present computer vision solutions to three related problems concerned with rodent tracking and observation. The first solution consists of a method to track rodents in a typical biomedical environment with minimal constraints. The method consists of two phases. In the first phase, a sliding window technique based on three features is used to track the rodent and determine its coarse position in the frame. The second phase uses the edge map and a system of pulses to fit the boundaries of the tracking window to the contour of the rodent. This solution presents two contributions. The first contribution consists of a new feature, the Overlapped Histograms of Intensity (OHI). The second contribution consists of a new segmentation method that uses an online edge-based background subtraction to segment the edges of the rodent. The proposed solution tracking accuracy is stable when applied to rodents with different sizes. It is also shown that the solution achieves better results than a state of the art tracking algorithm.

The second solution consists of a method to detect and identify three behaviors in rodents under typical biomedical conditions. The solution uses a rule-based method combined with a Multiple Classifier System (MCS) to detect and classify rearing, exploring and being static. The solution offers two contributions. The first contribution is a new method to detect rodent behavior using the Motion History Image (MHI). The second contribution is a new fusion rule to combine the estimations of several Support Vector Machine (SVM) Classifiers. The solution achieves an 87% recognition accuracy rate. This is compliant with typical requirements

in biomedical research. The solution also compares favorably to other state of the art solutions.

The third solution comprises a tracking algorithm that has the same apparent behavior and that maintains the robustness of the CONDENSATION algorithm. The tracking algorithm simplifies the operations and reduces the computational load of the CONDENSATION algorithm while conserving similar tracking accuracy. The solution contributes to a new scheme to assign the particles at a certain time step to the particles of the previous time step. This scheme reduces the number of complex operations required by the classic CONDENSATION algorithm. The solution also contributes a method to reduce the average number of particles generated at each time step, while maintaining the same maximum number of particles as in the classic CONDENSATION algorithm. Finally, the solution achieves 4.4× to 12× acceleration when compared to the classical CONDENSATION algorithm, while maintaining roughly the same tracking accuracy.

RÉSUMÉ

Les rongeurs sont régulièrement utilisés dans les expériences et la recherche biomédicale. Ceci est dû entre autres aux caractéristiques qu'ils partagent avec les humains, au faible coût et la facilité de leur entretien, et à la brièveté de leur cycle de vie.

La recherche sur les rongeurs implique généralement de longues périodes de surveillance et de suivi. Quand cela est fait manuellement, ces tâches sont très fastidieuses et possiblement erronées. Ces tâches impliquent un technicien pour noter la position ou le comportement du rongeur en chaque instant. Des solutions de surveillance et de suivi automatique ont été mises au point pour diminuer la quantité de travail manuel et permettre de plus longues périodes de surveillance. Plusieurs des solutions proposées pour la surveillance automatique des animaux utilisent des capteurs mécaniques. Même si ces solutions ont été couronnées de succès dans leurs tâches prévues, les caméras vidéo sont toujours indispensables pour la validation ultérieure. Pour cette raison, il est logique d'utiliser la vision artificielle comme un moyen de surveiller et de suivre les rongeurs.

Dans cette thèse, nous présentons des solutions de vision artificielle à trois problèmes connexes concernant le suivi et l'observation de rongeurs. La première solution consiste en un procédé pour suivre les rongeurs dans un environnement biomédical typique avec des contraintes minimales. La méthode est faite de deux phases. Dans la première phase, une technique de fenêtre glissante fondée sur trois caractéristiques est utilisée pour suivre le rongeur et déterminer sa position approximative dans le cadre. La seconde phase utilise la carte d'arrêts et un système d'impulsions pour ajuster les limites de la fenêtre de suivi aux contours du rongeur. Cette solution présente deux contributions. La première contribution consiste en une nouvelle caractéristique, les histogrammes d'intensité qui se chevauchent. La seconde contribution consiste en un nouveau procédé de segmentation qui utilise une soustraction d'arrière-plan en ligne basée sur les arrêts pour segmenter les bords du rongeur. La précision de suivi de la solution proposée est stable lorsqu'elle est appliquée à des rongeurs de tailles différentes. Il est également montré que la solution permet d'obtenir de meilleurs résultats qu'une méthode de l'état d'art.

La deuxième solution consiste en un procédé pour détecter et identifier trois comportements chez les rongeurs dans des conditions biomédicales typiques. La solution utilise une méthode basée sur des règles combinée avec un système de classificateur multiple pour détecter et classifier le redressement, l'exploration et l'état statique chez un rongeur. La solution offre

deux contributions. La première contribution consiste en une nouvelle méthode pour détecter le comportement des rongeurs en utilisant l'image historique du mouvement. La seconde contribution est une nouvelle règle de fusion pour combiner les estimations de plusieurs classificateurs de machine à vecteur du support. La solution permet d'obtenir un taux de précision de reconnaissance de 87%. Ceci est conforme aux exigences typiques dans la recherche biomédicale. La solution se compare favorablement à d'autres solutions de l'état de l'art.

La troisième solution comprend un algorithme de suivi qui a le même comportement apparent et qui maintient la robustesse de l'algorithme de CONDENSATION. L'algorithme de suivi simplifie les opérations et réduit la charge de calcul de l'algorithme de CONDENSATION tandis qu'il maintient une précision de localisation semblable. La solution contribue à un nouveau dispositif pour attribuer les particules, à un certain intervalle de temps, aux particules du pas de temps précédent. Ce système réduit le nombre d'opérations complexes requis par l'algorithme de CONDENSATION classique. La solution contribue également à un procédé pour réduire le nombre moyen de particules générées au niveau de chaque pas de temps, tout en maintenant le même nombre maximal des particules comme dans l'algorithme de CONDENSATION classique. Finalement, la solution atteint une accélération 4,4 × à 12 × par rapport à l'algorithme de CONDENSATION classique, tout en conservant à peu près la même précision de suivi.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AALAS	American Association for Laboratory Animal Science
BRISK	Binary Robust Invariant Scalable Keypoints
CHR	Canadians for Health Research
CONDENSATIONh	CONDENSATION using the histogram of intensities as observation model
DBN	Dynamic Bayesian Network
DoG	Difference of Gaussians
DoOG	Difference of Offset Gaussians
e-background	Edge-Background
GLOH	Gradient Location-Orientation Histogram
HMM	Hidden Markov Model
HOF	Histograms of Optical Flow
HOG	Histograms of Oriented Gradients
HoI	Histogram of Intensities
KPF	Kalman Particle Filter
MBH	Motion Boundary Histograms
MCS	Multiple Classifier System
MEI	Motion Energy Image
MHI	Motion History Image
MHV	Motion History Volume
OHI	Overlapped Histograms of Intensities
PBT	Probabilistic Boosting-Tree
PCA	Principal Component Analysis
RBF	Radial Basis Function
SETA	Simplified Efficient Tracking Algorithm

SIFER	Scale-Invariant Feature Detector with Error Resilience
SIFT	Scale Invariant Features Transform
SSD	Sum of Squared Differences
SVM	Support Vector Machine
UPF	Unscented Particle Filter

CHAPTER 1 INTRODUCTION

1.1 Overview and motivation

According to the American Association for Laboratory Animal Science (AALAS) (2013) and to Canadians for Health Research (CHR) (2013), rodents represent 90% of all the animals used in biomedical research. Rodents are favored in biomedical research for several reasons. Both rodents and humans are mammals, and they share many similarities in structure, genetic attributes, behavior and organ functionalities (CHR, 2013). Consequently, rodents are susceptible to many common illnesses and diseases that affect humans, and many human symptoms can be replicated in rodents. Moreover, rodents are small in size, they are easy to handle, have low cost, are easy to house and maintain, and are able to breed in captivity (CHR, 2013). In fact, rodents in biomedical laboratories are specifically bred for research purposes (AALAS, 2013). Furthermore, rodents' life span is short (two to three years). This allows for experiments to be conducted on a complete life cycle or even several generations (AALAS, 2013). Researchers are also well familiar with rodent anatomy, physiology and genetics. This makes changes in rodent behavior or characteristics easy to detect and understand. Rodents have been used as models for a large number of human diseases and disorders, including hypertension, diabetes, obesity, respiratory problems, cancer, heart disease, aging and seizures (CHR, 2013) (AALAS, 2013). Our partners in CHU Saint-Justine research center are using rats as models to study the effect of seizures on human brains (Gibbs, S., et al., 2011).

Research on rodents involves long periods of tracking and monitoring, especially when the experiment extends over a long period of time. Animal tracking consists of observing the animal and recording its position at each instance. This is useful to draw a rodent's motion pattern and infer its mood and psychological state. For instance, when a rat is stressed, it tends to stay in dark and isolated places. Whereas, when the stress subsides the rodent is more likely to explore its environment (Ishii, H., et al., 2007). Monitoring consists of observing a rodent to detect and identify certain behaviors of interest. The behaviors include basic ones such as rearing, walking, and being inactive. They also include more complex behaviors like exploring, eating, drinking or having a seizure.

Manual rodent tracking and monitoring is a very tedious task that is prone to error. It involves a trained technician meticulously observing the rodent directly or through pre-recorded videos for long durations that could extend over several hours. The technician annotates the target behavior or location at each time step. Given the enormous amount of manual work and time

that this task requires, processing observations over such long durations becomes impractical. For this reason, in some experiments the observation durations are abbreviated to produce results in a reasonable time. For example, Greer and Capecchi (2002) used three four-hours segments of video sequences to study the behavior of rodents affected with a disruption of Hoxb8, a protein found in a gene. However, reducing the observation time may lead to less accurate results and faulty conclusions.

Automatic tracking and monitoring solutions allow for longer monitoring periods which may extend to days. They monitor the rodents and provide analytical and condensed summaries for technicians to verify and to draw conclusions upon. These solutions can also reduce observation error as they are not subject to human fatigue and reduction in concentration. In addition, these solutions save technician time that can be invested in more productive tasks.

1.2 Problem Statement

In this thesis, we propose exploiting the power of computer vision to monitor and track rodents. This approach presents several important challenges. First, rodents have extremely deformable bodies. This makes them very hard to model. Except for some facial features like the eyes, the ears and the snout, no feature could be discernible.

It is an important consideration of this thesis to propose techniques that can be easily adapted by medical researchers. Consequently, we aim to introduce techniques and methods that require as little change as possible to existing laboratory environments. This implies using existing animal cages, in actual laboratory settings, without making significant changes to lighting or other environmental parameters.

A second important challenge therefore comes from the animal cages. During experiments, rodents are usually housed in Plexiglas cages to allow observation. It is common to have reflections on the Plexiglas surface, whether from lighting, objects or people present in the experimentation room. Reflections are a major cause of noise when using a computer vision technique.

Third, the cage floors are covered with bedding for the rodents' comfort. Bedding is highly textured and dynamic. This characteristic makes it difficult to model the bedding and constitutes another source of noise.

Fourth, in typical biomedical laboratories, lighting is seldom controlled. This causes a non-uniform illumination of the monitored cages. A non-uniform illumination tends to change the

color distribution of the rodent each time it crosses from one illumination zone to the next. This decreases the effectiveness of color modeling.

Hence, there is a need for a robust monitoring computer vision algorithm that achieves good results under such typical environmental constraints.

Furthermore, rodent may exhibit erratic motion. In such a case, when a particle filter such as the CONDENSATION algorithm is used for tracking, a large number of particles is needed to cover a wide search space and to achieve high tracking precision. The CONDENSATION algorithm also presents many complex operations, such as random value generation and floating point manipulation. These may also present several bottlenecks for a potential hardware implementation. They also cause high computational complexity and energy dissipation. Consequently, it is useful to have an efficient tracking algorithm that preserves the accuracy and the apparent functionalities of the CONDENSATION algorithm.

1.3 Research Objectives

The general goal of this thesis is to solve practical computer vision problems in rodent tracking and monitoring.

Given that tracking and monitoring rodents has mostly been done under controlled settings, we aim to develop an efficient and robust method that operates satisfactorily in typical biomedical laboratory environments and with minimal constraints. We assume minimal control over illumination and no disruption to cage localization. Typical transparent cages will be used to allow visibility and their content will be undisturbed. The only constraint is that each cage contains a single rodent.

As a separate but related issue, we also aim to track rodents and other subjects with a computer vision algorithm that preserves the apparent behavior and robustness of the CONDENSATION algorithm, while simplifying and adapting its operations to allow for an efficient possible hardware implementation.

The detailed objectives of the research presented in this thesis are:

1. We aim to find robust models to represent rodents for tracking and for behavior detection purposes taking into consideration the scarcity of physical features that are found

on the body of a rodent. We also should take into consideration high deformability of the rodent body.

2. We aim to find a robust and precise motion model to represent the displacement of rodent from one frame to the next frame.
3. We aim to find a method to extract the rodent from the rest of the frame and determine its boundaries.
4. We aim to find a method to detect and distinguish three behaviors (exploring, rearing, static) in rodents under those conditions. Those three behaviors were selected among a list of typical behaviors used to infer information about the state and health of the rodent in biomedical experiments.
5. We aim to find an efficient rodent tracking method that has the same apparent behavior and accuracy as the CONDENSATION algorithm.

1.4 Summary of Contributions

To solve the difficulties brought in by practical conditions when monitoring rodents, we have documented and proposed solutions in three journal papers. The main contributions are summarized in the following items.

Robust Rodent Tracking: Our contributions revolve around developing a robust rodent tracking and extracting algorithm that achieves adequate accuracy under a typical biomedical environment with minimal constraints.

1. We introduce a new feature: the Overlapped Histograms of Intensity (OHI). The OHI is a feature is similar to the Histogram of Oriented Gradients (HOG) but that use the intensity of the region of interest instead of the gradients. To calculate the OHI, the region of interest is divided according to a grid. The histograms of overlapping combinations of neighbouring grid cells are calculated. The histograms values are grouped together to form the resulting feature.
2. We propose a new segmentation method to extract the boundaries of the target using an online edge-background (e-backgound) subtraction method and edglet-based constructed pulses. Edglets are discontinuous fractions of edges and the e-background is a continuously updated accumulation of background edges.

Rodent Monitoring: Our contributions to rodent monitoring aim to yield a robust algorithm that detects and distinguishes exploring, rearing and being stationary in rodents with an accuracy that is sufficient for biomedical research under minimal environmental and experimental constraints.

1. We propose a new method to detect and distinguish exploring, rearing and being stationary in rodents using Motion History Images (MHIs).
2. We propose a new fusion rule that combines the estimations of several Support Vector Machines (SVMs). A fusion rule is an equation that is used to determine the contributions of each base classifier. This particular fusion rule is used for its simplicity and its ability to favor the classifier with the strongest estimation at each call.

Hardware efficient rodent tracking algorithm: Our contributions in this aspect revolve around proposing a tracking algorithm that reduces the computational complexity of the CONDENSATION algorithm without reducing its tracking precision and efficiency. We also propose a method to dynamically reduce the number of particle computed at each time step, hence reducing the computations and processing time.

1. The first contribution consists of proposing a new method for sampling that reduces time complexity while maintaining adequate precision.
2. The second contribution consists of a simplified method to assign the number of particles generated at a certain time step from the particles at the previous time step.

1.5 Thesis Structure

In chapter 2, we present a critical review of literature review along with a summary of the state of the art on rodent tracking and monitoring. In chapter 3, we present an overview of the methods presented in the thesis. In chapter 4, we present a robust rodent tracking and extracting algorithm in an article entitled “Catching a Rat by its Edglets” that was published in the *IEEE Transactions on Image Processing*. Chapter 5 presents a robust method to detect exploring, rearing and static in rodents that is detailed in an article entitled “Computing a Rat’s Diary” submitted to the *IEEE Transactions on Image Processing*. In Chapter 5, we present a tracking algorithm that allows for an efficient hardware implementation. The algorithm is described in an article entitled “A Computationally Efficient Importance Sampling Tracking Algorithm” that was submitted to the *Machine Vision and Applications* journal. An earlier version of this work was published in the *International Conference on Electrical and Com-*

puter Systems in 2011. Chapter 6 presents a general discussion of the different aspects of our research. Finally, chapter 7 concludes the thesis by summarizing our contributions and outlining future research directions.

CHAPTER 2 LITERATURE REVIEW

In this chapter, we review the literature relevant to the problem of tracking rodents and monitoring their behavior in a medical context. First, we review commercial systems that exploit mechanical sensors. Second, we review computer vision solutions for video tracking. Third, we consider computer vision approaches to track rodents. Fourth, we review methods for behavior detection. Finally, we consider works that exploit computer vision approaches to monitor rodents.

2.1 Systems based on mechanical sensing

Several solutions have been provided for automatic rodent monitoring. These systems rely on different types of sensors that include vibration sensors, photo sensors and cameras. Systems equipped with cameras will be discussed in later sections.

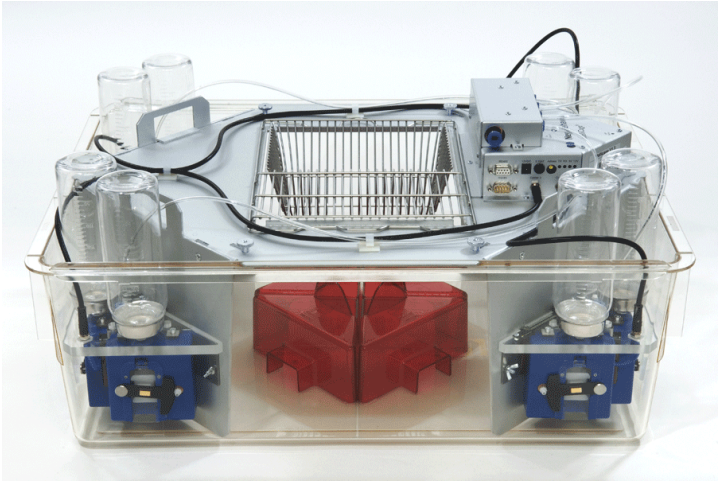


Figure 2.1 IntelliCage By NewBehavior (Courtesy (IntelliCage, 2013))

NewBehavior provides an industrial system, IntelliCage that detects whether the rodent is present in one of the four instrumented corners of the cage, and if the rodent's snout pokes the water bottle in one of those corners (New Behavior, 2013). The corners are equipped with individual transponders and temperature sensors to detect the presence of the rodent (see Figure 2.1). They are also equiped with light beams and photosensors installed at the doors of the water feeders. A nose poke interrupts the light beam and allows access to the water bottle.

Though the system serves its objectives of monitoring the feeding of a rodent, it is incapable of providing any additional information on other behaviors, especially when the rodent is not in one of the four instrumented corners.

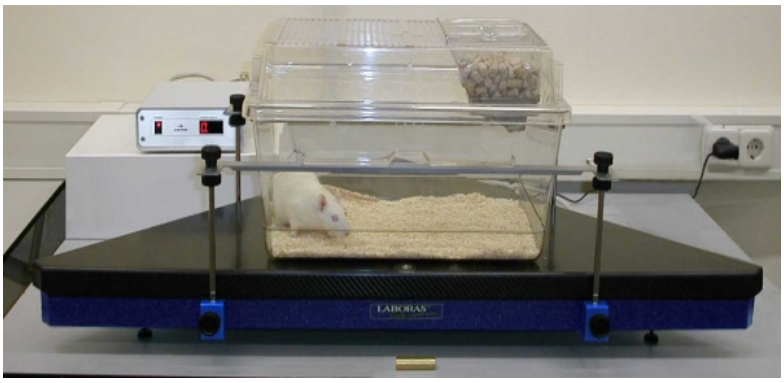


Figure 2.2 The Laboras system: a transparent cage is placed on a heavy baseplate equipped with vibration sensors (Courtesy of (Metris, 2013))

Metris provides another product for laboratory rodents, Laboras, which detects vibrations to identify behaviors such as resting, grooming, eating, drinking, locomotion and climbing (Laboras, 2013). The system consists of a heavy baseplate in which force transducers are installed, a carbon fiber measurement plate that covers the transducers, and a construction to hold the cage (see Figure 2.2). The system measures, amplifies and analyses the vibrations caused by the rodent on the measurement plate and deduces the corresponding behavior.

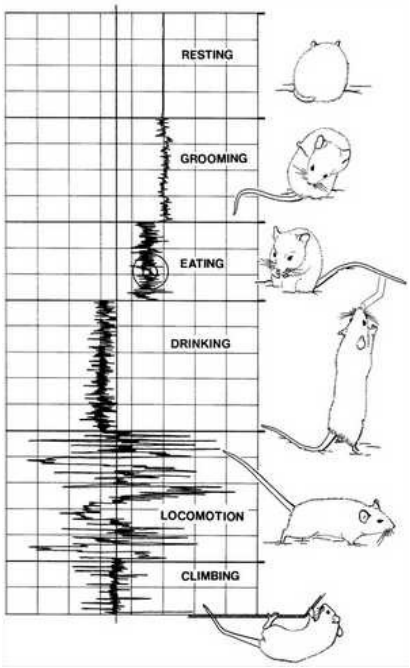


Figure 2.3 Behavior detection using Laboras (Courtesy of (Metris, 2013))

For example, as shown in Figure 2.3, when the system output is reduced to a straight line the rodent is assumed to be resting. When the system exhibits vibrations with very high amplitude, the rodent is assumed to be in locomotion.

Ishii et al. (2007) proposed a cage environment to count the number of grooming and rearing events. The cage was built from black-colored wood (see Figure 2.4) to facilitate image processing. The cage was equipped with food and water supply machine. To determine grooming, four vibration sensors were positioned in the four corners of the cage. The assumption is that a rodent seeks a narrow place to groom, and will thus tend to migrate to the cage corners when adopting this behaviour. During grooming, a rat shakes its head at a certain constant frequency that ranges between 3 Hz and 6 Hz. This motion creates vibrations that propagate to the floor and can then be picked up by the vibration sensors. To detect grooming, it suffices to isolate vibrations at a given frequency and amplitude. Photo interrupters are placed near the feeding and drinking areas to detect when the rodent is approaching them. Rearing is also detected by photo interrupters positioned, on the cage wall, at 110 mm from the floor of the cage. This is justified by a threshold that separates the height of a rat standing on four feet and a rat standing on two feet.

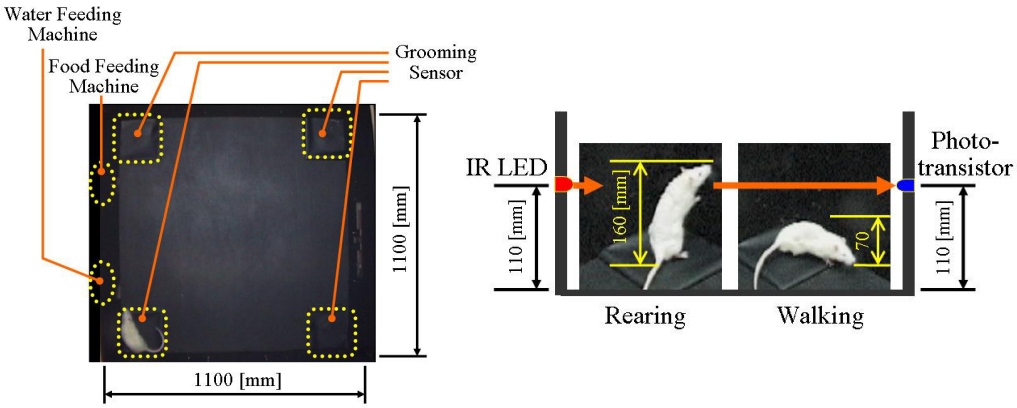


Figure 2.4 Details of the system described in (Ishii H., et al., 2007). The cage is equiped with a water feeding machine, food feed machine, photosensors and four grooming sensors in the corners (©2007, IEEE).

Jackson et al. (2003) used a system of projected infrared beams in a cage. Six evenly spaced beams were installed near the floor of the cage and two were installed at mid-height. The rodent is assumed to be roaming, rearing, or climbing depending on the number and location of the infrared beams broken, and for the duration of the breaks. For instance, a rodent is as-

sumed rearing or climbing if one upper beam is broken. A rodent is assumed to be roaming if it is continuously breaking at least three lower beams for two minutes at a time.

Although these systems are efficient, they are limited in the types of behaviors and states that one sensor can detect. For instance, vibration sensors, as in (Ishii H., et al., 2007) can only detect grooming. To track the rodent, video processing was used. Light sensors as in (Jackson, W. S., et al. 2003) can be used to detect the location of the rodent at each time instance and consequently track the rodent. Light sensors are sensors that are sensitive to certain light frequencies. Light sensors can also be used to determine the posture of the rodent. A light beam of that frequency is pointed at the sensor at all times. Whenever the beam is broken by a moving object that crosses the line of sight between the light source and the light sensor, the sensor emits a signal.

However, light sensors cannot provide any additional information about the behavior of the rodent such as if the rodent is grooming or scratching. These sensors are even more limited, since they can only provide information about the position of the rodent (New Behavior, 2013).

To detect several kinds of behaviors, a combination of different types of sensors should be used. This will clutter the cage space. Furthermore, even though those sensors are used for automatic detection of the rodent behavior, a video camera is still often used to record the experiment for future validation by a human expert.

Computer vision is a possible alternative tool. Computer vision can be used to detect all the behaviors detected by any of the sensors discussed previously as it will be demonstrated in section 2.5. This makes video cameras universal sensors. HomeCageScan, an industrial product from CleverSys (Wilks, S.L., et al., 2004; Liang, Y., et al., 2007), detects twenty-one behaviors that include all the ones covered by the described sensors, using only computer vision techniques. In addition, given that video cameras are indispensable equipment for expert validation, it is more convenient to use them also for automatic monitoring. This reduces the additional instrumentation that surrounds the cage environment and possibly the cost.

2.2 Video Tracking Techniques

Video tracking is a process of following certain image elements of interest throughout a video sequence. In this section, we classify video tracking techniques according to the system proposed by Trucco and Plakas (2006).

Tracking systems consist of two basic elements: motion and matching. The motion element task is to identify a narrow region of interest where the target has a high probability to exist. The matching element consists of identifying the target in the region of interest. Trucco and Plakas proposed a video tracking algorithm classification system based on target model complexity. They identified six classes: window tracking, feature tracking, tracking using planar rigid shapes, tracking using solid rigid objects, tracking deformable contours, and tracking using visual learning.

Window tracking consists of tracking a simple rectangular region across the video sequence. It is also based on the assumption that the intensity patterns do not present large differences between two consecutive frames. This assumption is necessary to establish the similarity between two windows that belong to the same objects in two consecutive frames. For instance, consider the case illustrated in (Trucco & Verri, 1998), with two consecutive frames F_t and F_{t+1} at time t and $t+1$ and two pixels P_t, P_{t+1} in F_t and F_{t+1}, respectively. Given $2W+1$ as the width and height of the tracking window, $R(p_t)$ the search region in F_{t+1}, and $\psi(u,v)$ a function that measures the similarity between two pixels u and v, the position of the tracking window in F_{t+1} is calculated according to the translation vector $\bar{d} = [d_1, d_2]$ that maximizes $c(d)$

$$c(d) = \sum_{k=-W}^{W} \sum_{l=-W}^{W} \psi(I_t(i+k, j+l), I_{t+1}(i+k-d_1, j+l-d_2)), \qquad (2.1)$$

where $I_t(i,j)$ and $I_{t+1}(i,j)$ are the intensities of the pixels situated at the coordinates (i,j) in $F_t$and F_{t+1} respectively.

There are multiple choices for $\psi(u,v)$, such as a cross correlation function or a Sum of Squared Differences (SSD) function.

In the case of a tracking window, the region of interest in F_{t+1} may be defined as the region that surrounds the target in F_t. To return accurate tracking results when using tracking windows, the intensity pattern is assumed to undergo small changes.

Feature tracking consists of first locating image elements, or features, in two consecutive frames and then matching each feature of the first frame to a feature in the second frame assuming a match exists. Feature tracking algorithms could be further divided into three categories: tracking local features, optical flow, and tracking extended features.

Local features are elements in the image such as edges and corners Moravec (1979). According to Moravec (1979) a corner is a point around which the intensity changes on different directions. The Scale Invariant Features Transform (SIFT) are also local features (Lowe, 2004). SIFT features are weighted histograms of gradient orientation that are computed locally around keypoints. Keypoints are calculated maxima and minima in different scale Difference of Gaussian (DoG) frames. The DoG was first introduced by Wilson & Giese (1977). It is a frame that results from the difference of two versions of the same frame. Each version is convolved with a Gaussian kernel having a different standard deviation.

Later on, Mikolajczyk and Schmid (2005) proposed the Gradient Location-Orientation Histogram (GLOH) that is an extension of SIFT. For GLOH, SIFT is computed using a log-polar histogram instead of a regular histogram. SIFT further evolved into several local descriptors among them PCA-SIFT (Ke & Sukthankar, 2004) that reduces the dimension of the SIFT feature using PCA. Mainali et. al (2013) extended SIFT by applying Cosine Modulated Gaussian filter instead of regular Gaussian filters to compute the DoG. They dubbed their descriptor Scale-Invariant Feature Detector with Error Resilience (SIFER). Leutenegger et al. (2011) proposed a binary like SIFT feature that they labelled Binary Robust Invariant Scalable Keypoints (BRISK). The feature mainly was used to simplify and reduce the computation load required by SIFT.

To track local features such as SIFT features, Smith et al. (1998), stated that features are matched by evaluating the similarities between the feature regions in two consecutive frames and pairing the features that have the highest similarities. Similarity may be tested using standard cross-correlation

$$standard\_cross\_correlation = \frac{\sum ij}{\sqrt{\sum i^2 \sum j^2}}, \tag{2.2}$$

where i and j are two values that belong to the two features to be compared, respectively.

Similarities may also be tested using a zero mean cross-correlation

$$zero\_mean\_cross\_correlation = \frac{\sum(i-\bar{\imath})(j-\bar{\jmath})}{\sqrt{\sum i^2 \sum j^2}}, \tag{2.3}$$

where $\bar{\imath}$ and $\bar{\jmath}$ are the means of the values attributed to characteristics of the pixels under considerations in each of the regions respectively.

The SSD is also used to measure the similarity between two regions

$$SSD = \sum(i - j)^2 \ , \tag{2.4}$$

The Euclidean distance is another common way to measure the similarities between two regions

$$euclidean\_distance = \sqrt{\sum(i - j)^2} \ , \tag{2.5}$$

The main difference between local features methods and window tracking methods is that for the local features methods, the features are calculated around interest points spread on the complete frame. In the window tracking method, the features are calculated inside a window selected at a certain position on the frame.

KLT (Kanade, Lucas, Tomasi) is a tracking method based on optical flow. Optical flow methods are based on the assumption that local brightness in a given region is constant and a change in brightness implies motion. For this assumption to hold, illumination is assumed to be uniform over the whole frame. Under those two assumptions, the brightness of an object is only due to its reflectance. In such cases, a change in brightness implies motion.

Lucas and Kanade (1981) proposed image registration by observing optical flow in two corresponding images. The same method is used in video tracking by tracing the displacement of gradient based features in two consecutive frames. The method assumes that the target undergoes a certain transformation (usually a translation) between the two frames. For this reason, it applies a transformation on the second frame, and tries to minimize the dissimilarity between the two frames by iteratively changing the parameter values of the transformation. Later state of the art methods tried to accelerate the algorithm or improve on some of the approximations that Lucas and Kanade (1981) made. Such methods are described in (Shum & Szeliski, 2000; Baker & Mathews, 2001; Hager & Belhumeur, 1998).

Rodent exhibit sudden movement and their motion and structure cause high variations in their texture and color distributions and their silhouette shape. Methods that assume small variations between two consecutive frames would not be efficient in this case. Such methods include techniques that use window tracking and feature tracking.

Extended features are features that cover larger regions in a frame than local features. Extended features may be contours of basic shapes including rectangles and ellipses. They may also be free-form contours or image regions. Image regions are defined as the connected parts of

an image that share similar properties such as color, texture or their dissimilarity to the background.

Fuentes and Velastin (2006) used an extended feature to track people in a metro station. The authors relied on a blob extraction method that they proposed in (Fuentes & Velastin, 2001) to extract human silhouette from the frames. Blobs are regions of interest in a frame that do not belong to the background. In this case, blobs are extracted by measuring the luminance contrast C with respect to the background at each pixel in the YUV color space. The blobs are then enclosed in bounding boxes. Tracking is achieved by matching each two overlapping boxes in two consecutive frames.

The mean shift tracker is also an extended feature tracking method. As outlined by Comaniciu et al. (2003), the mean shift tracker uses an ellipse to model the target and the color distribution as the attribute that characterizes the target. The mean shift tracker proceeds as follows:

1. Initialize the algorithm by drawing an ellipse around the target in frame F_1 at $t = 1$.
2. In frame F_t at time t, calculate the color histogram of the region of the frame enclosed by the ellipse and calculate the mean of the pixels according to equation 2.6.

$$y_0 = \frac{\sum_{i=1}^{n_h} x_i w_i}{\sum_{i=1}^{n_h} w_i}, \tag{2.6}$$

   where y_0 is the centroid of the target, x_i and w_i are the coordinates and the weight assigned to a pixel respectively.
3. In frame $F_{t=1}$ at $t + 1$, set the center of the target $\hat{y}_0 = y_0$.
4. Calculate the color histogram of the region enclosed by the ellipse centered at $\hat{y}_0$ and the new mean of the pixels $\hat{y}_1$ according to equation 2.6.
5. If the difference between $\hat{y}_0$ and $\hat{y}_1$ is less than ϵ, then the center of the target at time $t + 1$ is set to $\hat{y}_1$.
6. Otherwise, set $\hat{y}_0 = \hat{y}_1$ and repeat steps 4 and 5 till the difference between $\hat{y}_0$ and $\hat{y}_1$ falls below ϵ, or that the number of iterations reaches a certain N_{max}.

Comaniciu et al. (2003) used the mean-shift algorithm to track people. The tracker proved robust even when tested on low quality recordings. It also proved robust to changes of appearance and scale. However, the mean-shift tracker performed poorly under fast changes and occlusions.

Planar rigid shapes and solid rigid object-based tracking methods assume that targets have unique characteristics and can be modeled in simple 2D or 3D shapes. The tracking problem is reduced in this case to a minimization problem. It minimizes the difference between a template model and an image region. Wunsh and Hirzinger (1997) used a solid rigid object based tracking method to track object. Contours are extracted from the current frame. These contours are matched with the contours of a 3D template model by minimizing a certain error metric, taking into consideration rotation and translation. Planar rigid shapes based tracking methods are not independent of the position and orientation of the camera given that the model of the target may change with different point of views.

Deformable contours, also known as active contours or snakes, are used for tracking. Snakes are spline curves with control points whose purpose is to find the outer edges of a target and to attach to it. Snakes are controlled by two forms of energy, the image energy and outer-energy (Kass, M. et al., 1988). The image energy tends to draw the snake towards image elements such as edges and corners, while the external energy tends to ensure the smoothness of the curve and limit its shape transformations. Snakes attach to their target by minimizing the overall energy function. Every time the target moves, the snake strives to maintain its connection to it by minimizing the overall energy again, thus tracking the target. However, snakes are easily distracted by image noise located near the target. The snake may lock on nearby noise and lose track of the actual target (Hao and Drew, 2002).

The environment in which rodents are usually kept in a biomedical environment presents a high degree of noise due to reflections which are mainly present on the plexiglass and metallic material of the cages and due to the highly dynamic and textured bedding. Accordingly, techniques such as active contours that are vulnerable to noise would be distracted by the omnipresent noise in the frame and fail to track the rodents.

Visual learning techniques consist of learning the shape and dynamics of the target through training. For example, a large dataset of images or sequences representing a target's appearance or dynamics is used to infer a model of the target using a technique such as Principal Component Analysis (PCA). Statistical learning using classifiers such as Support Vector Machines (SVMs) can then be applied. For instance, Baumberg and Hogg (1994) used PCA to represent objects. The objective of PCA is to build a low-dimensional and flexible model of a complex target. Judd et al. (2009) collected several features such as the distance of each pixel from the center of the frame, a face location of a person in a frame, a human location in a frame, horizontal lines and the probability of the red, green and blue channels to train an SVM

classifier to track the gaze of a person. Kratz and Nishino (2010), used a collection of HMM based on Local Spacio-Temporal Motion Patterns to track people in extremely Crowded Scenes. In a more recent work, Park et al. (2012) used an Autoregression HMM to track targets using Haar like features. Avidan (2007) used an AdaBoost classifier to track people. The tracker is based on two types of local features: the histograms of orientation and RGB color values of the pixels. AdaBoost classifiers will be reviewed on more details in Section 2.3.

Finally, the motion element may be processed using several techniques. Motion element is the element that identifies the search region. One technique, as seen above, is to scan the nearby region to the target at the previous frame. Trucco and Plakas (2006) also mentioned that this may be done using filters such as the Kalman filter (Blake, A. et al., 1993) and the particle filter (Isard & Blake, 1996).

The CONDENSATION algorithm, also known as the particle filter, was first used for a computer vision application by Isard and Blake (1996). It is illustrated in Figure 2.5. The authors used spline curves as the observation models to track objects such as hands and faces. However, the CONDENSATION algorithm is a framework that may be applied with different observation models, weights and templates.

The CONDENSATION algorithm is based on factored sampling. Given $\boldsymbol{v}_t$, the state of the target model at time *t*, $\mathcal{V}_t = \{\boldsymbol{v}_1, \dots, \boldsymbol{v}_t\}$ its history, $\boldsymbol{z}_t$ the observation (frame data) at time *t* and its history $\mathcal{Z}_t = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_t\}$, the conditional statement density $p(\boldsymbol{v}_t | \mathcal{Z}_t)$ is approximated by a sample set denoted by $\{\boldsymbol{s}_t^{(n)}, n = 1, \dots, N\}$ with weights $\pi_t^{(n)}$. The higher the value of *N* the more accurate the approximation is. $p(\boldsymbol{v}_t | \mathcal{Z}_t)$ is obtained through a series of sampling prediction and measurement. First a new sample set $\{\boldsymbol{s'}_{t+1}^{(n)}, n = 1, \dots, N\}$ is sampled from $\{\boldsymbol{s}_t^{(n)}, n = 1, \dots, N\}$. The sampling is biased by the weights $\pi_t^{(n)}$of each $\boldsymbol{s}_t^{(n)}$ and some samples could be selected several times while others could never be selected. Then, prediction is performed by propagating the samples using a linear transformation. The linear transformation is formed of two components: a drift and a diffusion. The drift consists of a deterministic transformation applied to a sample. When a drift is applied to two samples with the same origin, the samples remain identical. A diffusion consists of affecting the samples with a random parameter. The drift and the diffusion are illustrated in equation 2.7.

$$\boldsymbol{s}_{t+1}^{(n)} = A\boldsymbol{s'}_t^{(n)} + B\boldsymbol{w}_t^{(n)}\,, \tag{2.7}$$

where $\boldsymbol{s}_{t+1}^{(n)}$ is a sample at t+1, A is the drift parameter, B the diffusion parameter and $\boldsymbol{w}_{t}^{(n)}$ is a random variable that belongs to a Gaussian distribution. The samples are then measured and weighted with respect to the observation $\boldsymbol{z}_{t+1}$ according to equation 2.8.

$$\pi_{t+1}^{(n)} = p(\boldsymbol{z}_{t+1}|\boldsymbol{x}_{t+1} = \boldsymbol{s}_{t+1}^{(n)}) \,, \tag{2.8}$$

In other words, using the observation, the weight assigned to a sample $\boldsymbol{s}_{t+1}^{(n)}$ reflects its likelihood to be the actual target given the observation $\boldsymbol{z}_{t+1}$.

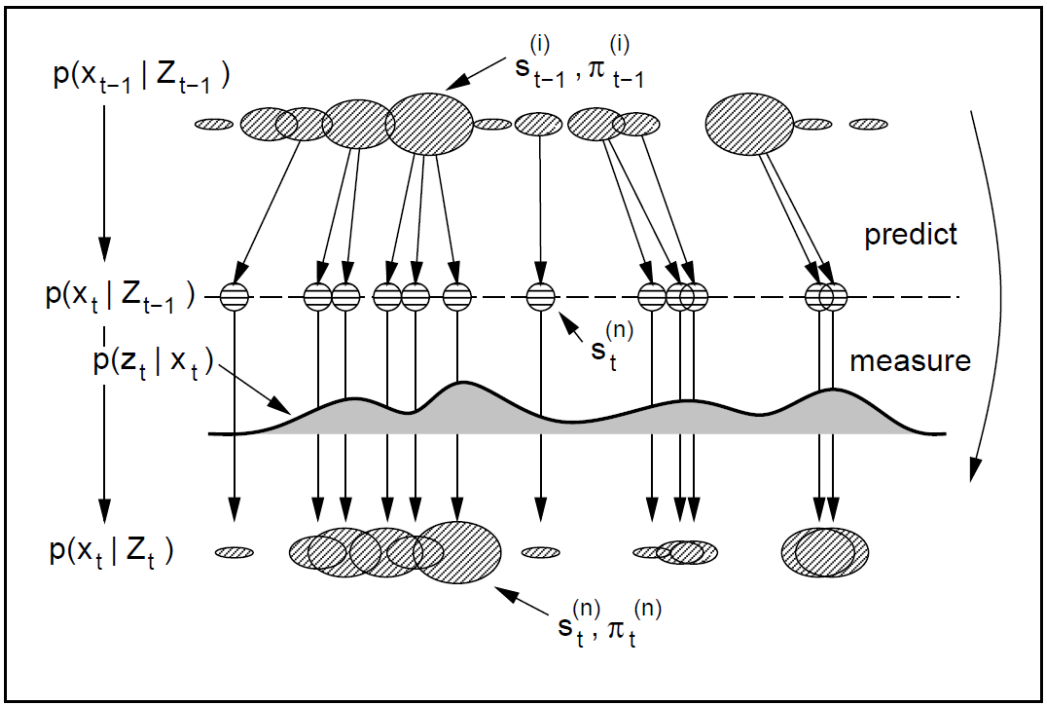


Figure 2.5 One time step in the CONDENSATION algorithm. Blob centers represent sample values and sizes depict sample weights (© 1998, IEEE).

2.3 Rodent Tracking and Extraction

In this section, we review state of the art methods for rodent tracking.

Pistori et al. (2010) and Goncalves et al. (2007) used a combination of a connected component based particle filter and the k-means algorithm to track several mice simultaneously. Binarization is used to extract the blobs, and then a particle filter is used to track the blobs of the mice (see Figure 2.6). The blobs of the mice are used to calculate the center of mass of each mouse and the dimensions of ellipses that approximate their contours. Binarization consists of applying a threshold to the frame intensity to segment the target blob. The centers of mass of

the ellipses are used to track the mice. When two or more mice come into contact, a switch is made to the k-means algorithm to separate the blob pixels in *k* groups to which the ellipses are fitted. The algorithm switches between the particle filter and the k-means algorithms based on a threshold that identifies the contact situation (Pistori, H., et al., 2010).

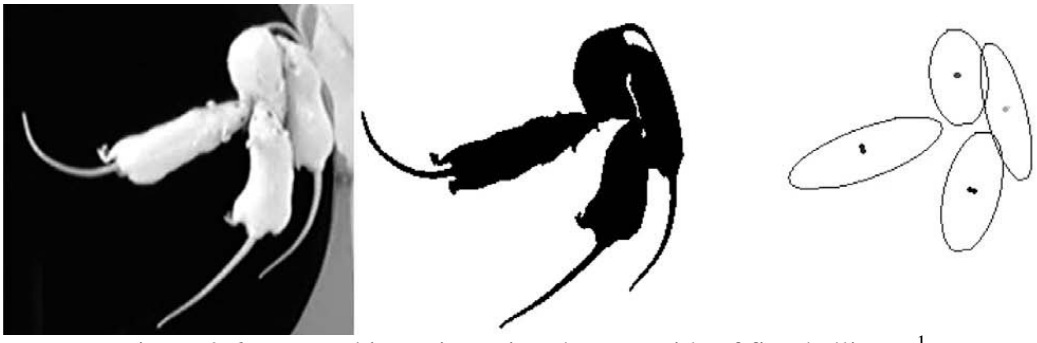


Figure 2.6 Tracking mice using the centroids of fitted ellipses [1]

Ishii et al. (2007) used computer vision to track the position of a mouse in a cage and determine its location. The video camera was placed above the cage and the system requires a black painted cage and a white mouse. The cage interior was entirely painted in black to ensure a high contrast between the mouse and its environment. Binarization was used to extract the mouse blob. The authors divided the cage's floor into a virtual grid to infer a distribution map of the mouse's position. This map was used to determine the rat's movement, its tendency to stay in certain locations in the cage and to deduce the mouse's emotional state. A rodent tends to stay in corners and narrow spaces when it is subjected to stress, and into the open when that stress factor disappears (Ishii, H., et al., 2007).

Nie et al. (2008, 2009) also used binarization to determine the location of a rat under two types of settings respectively. The first setting consisted of a high frame-rate camera to catch the rapid motion of the paws and a transparent cylindrical container. The camera was facing the setup. The container was filled with water, and one animal was placed at a time in the water. The second setup consisted of a transparent acrylic cage that contains a mouse. The cage was placed on an infrared flat illuminator which was a source of infrared radiation. The infrared illuminator was used to capture a clear silhouette of the mouse. The camera was placed on top of the cage. The experiment setup is shown in Figure 2.7.

1. Reprinted from Pattern Recognition Letters, 31, PISTORI, H., ODAKURA, V., MONTEIRO, J.B.O., GONÇALVES, W.N., ROEL, A. J., and MACHADO, B.B., Mice and larvae tracking using a particle filter with an auto-adjustable observation model, pp. 337–346., Copyright (2010), with permission from Elsevier

Using binarization requires a high contrast between the color of the target and the color of the background. This is not always possible in a practical environment. In typical biomedical laboratories, rodent cages are placed on shelves where the background is not controlled and can be cluttered with objects of different shapes, colors and textures. The color of the rodents is usually associated with its breed, and the breed of the animal is dictated by the ongoing experiment. The cages' floors are also usually covered with bedding to ensure the comfort of the rodent. In such case, the background and the bedding may share some of the rodent colors and the contrast constraint is not met. For example, our partners at St-Justine are studying Sprague-Dawley rats to remain consistent and to compare with previous results found at the laboratory. Sprague-Dawley rats are white. The animals are housed in Plexiglas cages stacked on shelves. The background consists mainly of the room's walls that are white. In this setting the contrast requirements are not respected and binarization cannot be used.

Figure 2.7 The set-up used by Nie et al. [2010] (© 2010, IEEE)

Brason and Belongie (2005) used a particle filter to track several mice. The particle filter used two models to represent its target, a blob and a contour. The algorithm uses the BraMBLE tracker (Isard & MacCormick, 2001) to segment a frame into blobs and background. It also uses the Berkley Segmentation Engine (BSE) (Martin, D. R., et al., 2004) to draw the edge map of each frame. The contour state is represented by an ellipse characterized by shape and velocity parameters. The contour state is represented by twelve contour templates equipped with measurement lines (see Figure 2.8). During sampling, the particle filter uses the similarity of the samples to the blob characteristics determined by the BraMBLE tracker template to weigh the samples. The algorithm also uses the degree of coincidence of the best template with the edge map to weight the samples again. The importance weight is computed as the

product of the blob contour weights. The authors mentioned that the selection of edge detection method is critical given that their method uses contour tracking. They also stressed the difficulty of isolating the mice edges among the clutter imposed by the environment. For instance, the mouse blends with the cage bedding and it is difficult to distinguish the mouse edges from the bedding edges. The authors stated that they chose BSE because it was the only edge detector that gave reasonable results.

Jhuang et al. (2010c) used background subtraction to track a dark rodent in a cage. The background consisted of the median of all the frames, then the background was subtracted from

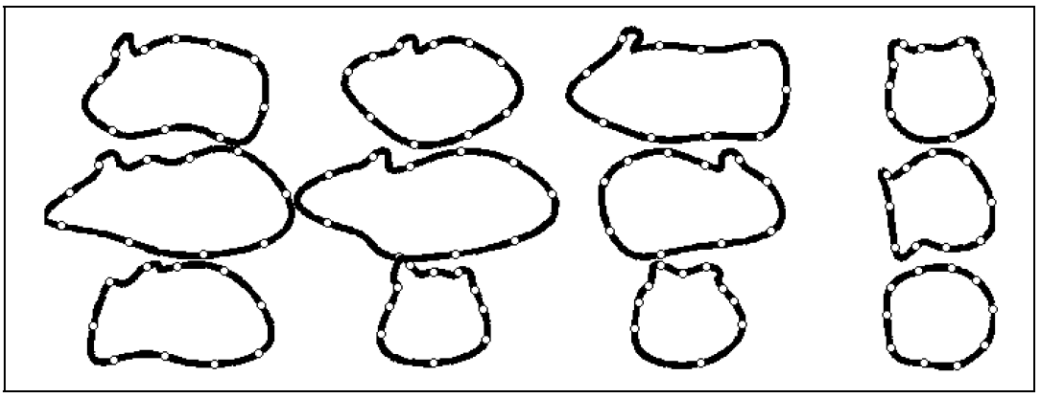


Figure 2.8 The 12 B-spline contour templates chosen to represent a mouse contour. The circles are locations of measurement lines (© 2005, IEEE).

each frame and the resulting blob was surrounded by a bounding box to indicate the boundaries of the rodent at each frame. The position of the bounding box indicated the position of the rodent at each frame, thus tracking the rodent.

Background subtraction is not advisable in uncontrolled environments for four reasons. First, the rodent may spend long durations sleeping in one location of the cage. In such a case, building the background using the frames of the video will result in a phantom shape of the rodent itself at the location where the rodent remains stationary. Second, the rodent size may fill considerable space in the cage resulting in large regions of the cage that are seldom visible. These regions cannot be represented in a standard color-based background. Third, if the background is constructed while the rodent is outside the cage, the background may get misaligned with the frames of the video when the rodent is later introduced in the cage. The person placing the rodent in the cage may displace it, or the rodent motion may displace the cage.

Fourth, the cage bedding is extremely dynamic. It is continuously displaced by the motion of the rodent.

Dollar et al. (2006) proposed to track rodent and other targets using a probabilistic edge detection method. Their method uses an extended Probabilistic Boosting-Tree (PBT) classifier (Tu, 2005) to identify the edges of the target by classifying a large number of features. The PBT classifier is a combination of a decision tree and boosting.

Boosting is a technique that combines several weak classifiers to form a strong classifier. A weak classifier has a classification accuracy that is close to a random decision (Viola & Jones, 2001). The general AdaBoost algorithm (Freund & Schapiro, 1997) combines a set of T weak classifiers h_t into a strong classifier H according to equation 2.9.

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t \,, \tag{2.9}$$

where x is a sample to be classified and α_ta weight attributed to h_t.

Tu (2005) used AdaBoost classifiers as building blocks for their proposed PBT classifier. The PBT is a decision tree whose nodes are strong classifiers. At each parent classifier, if the probability of a certain sample being a positive sample is greater than a given threshold, the sample is forwarded to the right side child node. If the probability of that sample being a negative sample is greater than another threshold, then the sample is forwarded to the left child node. Otherwise, the sample is forwarded to both child nodes. All the decisions of the descendant nodes are then sent back to the top parent node, where an overall decision is calculated based on those decisions. The same process is repeated recursively for each child node, until the tree end is reached. A positive sample is a sample that belongs to the target and a negative sample is a sample that does not.

Dollar et al. (2006) used PBT because it is fast in processing a large number of features. To train the system, manual segmentation is done on a training data set. The system then collects positive and negative features from the segmented frames for training. Features are collected at different scales and locations and they are categorized in three types: gradients, difference between histograms computed on Difference on Gaussians (DoG) responses or Difference of Offset Gaussians (DooG) responses and Haar wavelets (Haar, 1911). The DooG is a frame that results from the difference of the result of a given image convolved with a Gaussian kernel and a translation of the same filtered image.

Using manual selection in this case can be problematic. The barrier between the rodent and the cage bedding is difficult to distinguish. When manually drawing the contour of the target, parts of the bedding can be selected. This tends to affect the integrity of the training set and the accuracy of the segmentation results.

Given the reviewed literature on rodent tracking, we saw the need for a robust rodent tracking algorithm that is independent of the rodent color or size. We also noticed that using one feature is not enough for tracking in an uncontrolled environment. This urged us to propose a method that relies on several features to track rodents. In addition, we observed that a background segmentation based on color or a binarization is not sufficient to extract the blob of a rodent. For this purpose we propose a method that uses a combination of tracking and background subtraction in the edge map to segment the rodent and determine its position in each frame.

2.4 Behavior Detection and Classification techniques

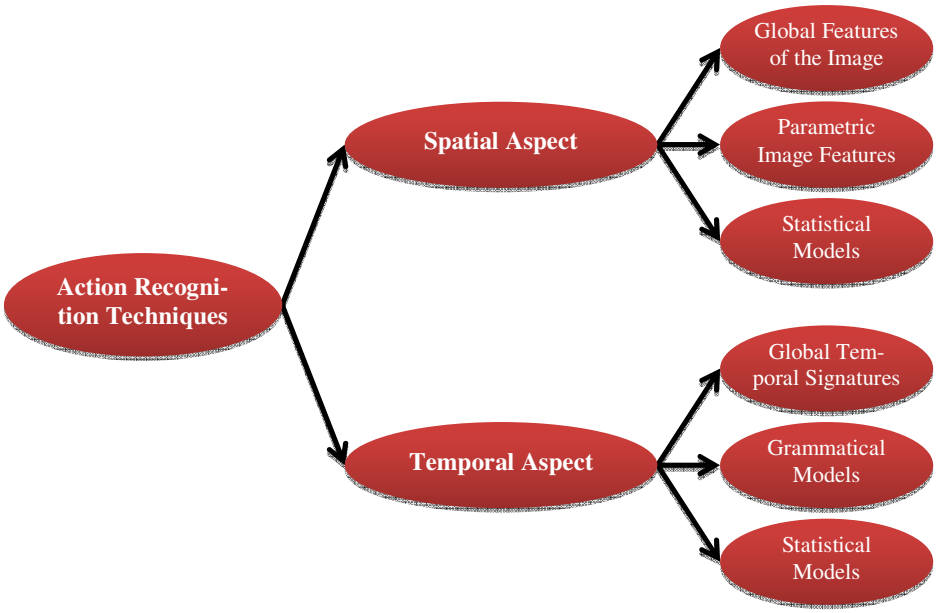


Figure 2.9 Action recognition techniques classification

In this section we briefly survey some state of the art behavior identification methods. We classify these methods according to the system proposed by Weinland et al. (2011).

Action recognition techniques may be categorized according to the spatial and temporal structures of the action (see Figure 2.9). Techniques that use the spatial aspect of the action may

further be divided into three categories: techniques that use global features of the image; techniques that use parametric image features; and techniques that use statistical models. Techniques that use the temporal aspect of the action may also be divided into three categories: techniques that use global temporal signatures; techniques that use grammatical models representing the sequential organization of the parts of an action and techniques that use statistical models.

Global image features are used to calculate the pose of the target. Furthermore, the pose of the target is used to recognize the action involved. For instance, it has been shown that human action recognition can be achieved merely by tracking the motion of specific points of the body of a person (Johansson, 1973). For this reason, computer vision techniques have relied on 3D (Ben-Arie, J., et al., 2002) and 2D (Guo, Y., et al., 1994; Jang-Hee, Y., et al., 2002) human models. Image features were used to fit the human models to the human image in the frame. Then, human action recognition was done by tracking and calculating the trajectories of certain anatomical parts of the human body such as the head or the limbs.

Image models consist of features collected on a region of interest (ROI) centered on the target. These features are then used to infer behaviors. Some classes of methods that use image model techniques proceed by extracting features from the target silhouette or the optical flow. For instance, Polana and Nelson (1992) computed temporal-textures. Temporal-textures consist of statistical values calculated using the direction and the magnitude of the optical flow. A nearest centroid classifier was used to classify the action. Cutler and Turk (1998) used the number, relative motion, size and position of optical flow blobs for action recognition. They then used a rule-based technique to determine the action based on the calculated features.

Bobick and Davis (2001) used the human silhouette to build Motion History Images (MHIs). The MHI consists of the accumulation of the human silhouette over several consecutive frames (see Figure 2.10).

The suggested method for action recognition using MHI proceeds by:

1. Using a background subtraction to extract the target silhouette.
2. Calculating the difference image Ψ between two consecutive frames to extract the motion pixels.
3. Calculating the MHI image according to equation 2.10

$$MHI(x,y) = \begin{cases} \tau & if\ \Psi(x,y) \neq 0 \\ 0 & else\ if\ MHI(x,y) < \tau - \delta \end{cases} \quad (2.10)$$

where (x, y) are the coordinates of a pixel in Ψ, τ is a timestamp, and δ is a decay factor.

4. Calculating the 7 Hu moments (Hu, 1962) of the MHI and grouping them in one feature vector.
5. Calculating the Mahalanobis distance (Therrien, 1989) from class templates to classify the processed action.

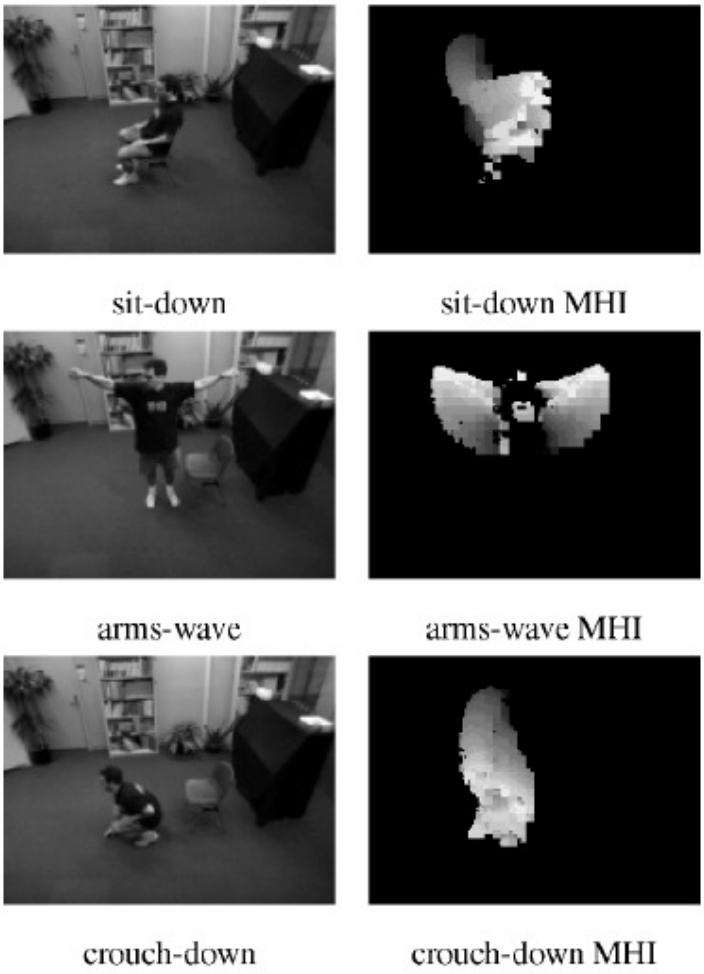


Figure 2.10 MHI for a person sitting down, waving their arms and person that crouches down (© 2001, IEEE).

Spatial statistics techniques base their action recognition on features collected from the entire frame, whereas, techniques that use image models use features collected on the ROI centered on the target.

For instance, Laptev (2005) extended the 2D Harris interest points for action recognition. The aim of the method is to extract spatio-temporal features from a video sequence that represent large changes in the spatial and temporal dimensions. These points are dubbed interest points. Interest points can be extracted using a second moment matrix integrated over a Gaussian window with independent spatial variance σ_i^2 and temporal variance τ_i^2. The result μ of this operation is shown in equation 2.11

$$\mu = g(x, y, t; \sigma_i^2, \tau_i^2) * \begin{pmatrix} L_x^2 & L_x L_y & L_x L_t \\ L_x L_y & L_y^2 & L_y L_t \\ L_x L_t & L_y L_t & L_t^2 \end{pmatrix}, \tag{2.11}$$

where $g(x, y, t; \sigma_i^2, \tau_i^2)$ is a Gaussian kernel, σ_i^2 and τ_i^2 are a spatial and temporal variance respectively. $*$ represents a convolution.

$$g(x, y, t; \sigma_i^2, \tau_i^2) = \frac{1}{\sqrt{(2\pi)^3 \sigma_i^4 \tau_i^4}} \times exp(-(x^2 + y^2)/2\sigma_i^2 - t^2/2\tau_i^2)\,, \tag{2.12}$$

and L_x, L_y, L_t are Gaussian derivatives computed at local scales σ_l^2 and τ_l^2

$$L_x = \delta_x(g(\cdot; \sigma_l^2, \tau_l^2) * f(\cdot)\,, \tag{2.13}$$

$$L_y = \delta_y(g(\cdot; \sigma_l^2, \tau_l^2) * f(\cdot)\,, \tag{2.14}$$

$$L_t = \delta_t(g(\cdot; \sigma_l^2, \tau_l^2) * f(\cdot)\,, \tag{2.15}$$

and $f(x, y, t)$ is the spatio-temporal image sequence.

Interest points are located at regions in f where the eigenvalues λ_1, λ_2, and λ_3 of μ have high values.

To classify actions taking place in the sequence, spatio-temporal feature vectors are formed out of the spatio-temporal Gaussian derivative of f at the interest points. K-means clustering is then used to detect similar events and classify them.

Histograms of Oriented Gradients (HOGs) are also spatial statistics features that are used for behavior recognition. They were first introduced by Dalal and Triggs (2005) for pedestrian recognition in scenes. To construct the features, the method decomposes the frame using overlapping cells. Then, for each cell a 1D vector is calculated. The vector represents the values of the orientation gradient histogram of the cell. The vectors calculated for all the cells are combined in one feature. These values are usually normalised. These features are then used to train a classifier and later detect and recognize behaviors in scenes. Several later algorithms extended the work done by Dalal and Triggs. For instance, Laptev et al. (2008), proposed 3D features formed out of HOG and Histograms of Optical Flow (HOF) features to detect actions in movie scenes. Dalal et al. (2006) introduced Motion Boundary Histograms (MBH). MBH are constructed out of the optical flow maps. The algorithm calculates the x and y optical flow maps. For each map, the orientation of gradient histograms are calculated in the same manner as for the HOG features. The resulting two features could be combined to form one resulting feature or they could be used separately in a winner-takes-it-all voting method.

Action grammar techniques represent actions as series of events that are ruled by transition functions. One of the most common techniques to classify actions using action grammar techniques are the Hidden Markov Models (HMMs) (Wang, T.-S., et al., 2001; Lv & Nevation, 2006). However, HMMs are sequential in their nature. This limits their ability to model certain types of actions, such as human actions in which several body parts may move simultaneously. To bypass this shortcoming, a technique such as the one proposed by Park and Agarwal (2003) and by Peursum et al. (2005) use Dynamic Bayesian Networks (DBN) to model more complex actions. Other techniques have also been used that are based on auto regressive models (Bissacco, A., et al., 2001), delayed neural Networks (Yang & Ahuja, 1999) and others.

Action template methods attempt to classify actions using templates of temporal blocks of features. These templates should not be confused with spatio-temporal features or with optical flow. Action template methods are computed over long sequence of frames compared to spatio-temporal features and optical flow that are computed on small time windows. Examples of action templates are Motion History Volumes (MHVs) (Weinland, D., et al., 2006). MHVs extend the 2D MHIs into 3D structures to represent view-invariant motion descriptors. The authors used several cameras to catch the motion in 3D. The MHV is then computed according to equation 2.16

$$MHV(x,y,z,t) = \begin{cases} \tau & if\ D(x,y,z,t) = 1 \\ \max(0, MHV(x,y,z,t-1) - 1) & otherwise \end{cases}, \quad (2.16)$$

Where $D(x,y,z,t)$ is the motion map.

Temporal statistical techniques are techniques that attempt to model the appearance of an action without considering its dynamic model. Such methods model actions with one characteristic frame labeled as the keyframe. For instance, Wang et al. (2006) used a keyframe method to identify certain human activities in images. The method consisted of sampling the edge map of an image and extracting 9x9 patches to describe the action in the image. Action matching is then done using clustering.

Other temporal statistics techniques used histograms of features to classify actions. These techniques are also known as temporal bag of features approaches. They proceed by extracting features from the sequence and building a histogram that represents the variance of these features. Action recognition is then reduced to classifying the histogram. Such a method is used by Scovanner et al. (2007).

2.5 Behavior Detection and Classification in Rodents

Rodent silhouette and behaviors are different from human silhouette and behaviors. For this reason, specific algorithms and methods have been proposed or adapted for rodent behavior detection and classification. In this section, we describe those techniques.

Nie et al. (2008) used a computer vision process to determine climbing, swimming and immobility in a forced swim test. A forced swim test is method that analyses the behavior of laboratory animals when placed in a cylindrical container filled with water. The technique required a high frame rate camera and a rule-based algorithm to discriminate the three behaviors. The method attempts to isolate and quantify the frequency of the vertical motion of the rat's body and paw motion. As shown in Table 2.1, the authors assumed that climbing is associated with a paw motion and a vertical motion. Swimming is associated with only a paw motion. Immobility is associated with the absence of both. Taking into consideration the assumptions outlined in Table 2.1 and the results from processing the sequence frames, the authors identified the behavior in every frame.

Table 2.1 Difference between immobility, climbing, and swimming (× represents the absence of the feature, ○ represents it presence).

	Paws' movement	**Body's vertical movement**
Immobility	×	×
Climbing	○	○
Swimming	○	×

To isolate and quantify the paw movement and the body's vertical movement, the technique proceeded as follows. The method uses binarization followed by frame differencing to extract and quantify motion pixels. At this stage, immobility may be detected if the number of motion pixels is smaller than a certain threshold. To detect climbing, the technique uses a morphological opening to delete the paws and the tail parts of the rodent. The purpose of the opening is to suppress the quick movement attributed to the paws. Then, the horizontal centroid position $v(t)$ is calculated for each frame according to equation 2.17

$$v(t) = \frac{\sum_x \sum_y y.D(x,y,t)}{\sum_x \sum_y D(x,y,t)}, \tag{2.17}$$

where (x, y) are the coordinates of a pixel at time t and $D(x, y, t)$ is the frame calculated after applying the opening.

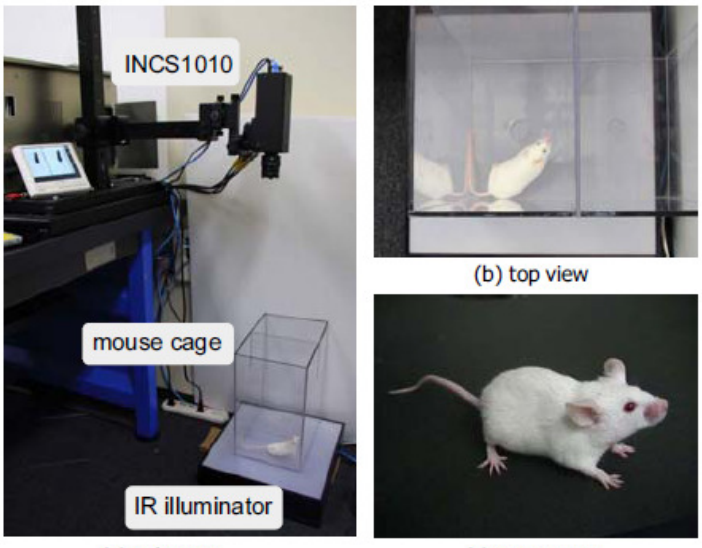


Figure 2.11 Experimental setting used in Nie et al. [2011] (© 2011, IEEE).

High pass filtering is then applied to $v(t)$ to extract the high frequency component in climbing according to equation 2.18

$$c(t) = \left(\left|H_f(v(t))\right|\right), \tag{2.18}$$

where $H_f(\cdot)$ is a high pass filter and f is the cutoff frequency. The calculation of $c(t)$ is illustrated in Figure 2- 11. Climbing may be isolated at this stage by thresholding $c(t)$. A high frequency vertical body motion (a large $c(t)$) indicates climbing. The remaining frames are reserved for swimming.

Nie et al. (2011) also used a high frame rate camera to identify moving, rearing, immobility, head grooming, and scratching for a mouse. The system consists of a transparent acrylic cage that contains a mouse. The cage is placed on an infrared flat illuminator that is used to illuminate the cage and ensure the capture of a clear silhouette of the mouse. The camera is placed above the cage for top view capture (see Figure 2- 12).

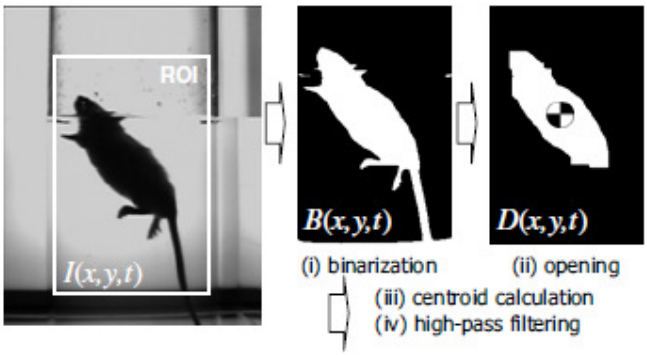


Figure 2.12 Calculating vertical body motion (© 2008, IEEE)

The authors first used binarization to extract the silhouette of the mouse and to calculate the coordinates of its centroid. Then, a Canny edge detector (Canny, 1986) was used to extract the edges of the rodent. The coordinates of the edges' pixels were transformed to their polar equivalent and their distance from the centroid was plotted versus the corresponding angle.

The interior contour of the rodent was formed of the pixels that belong to a certain angle and have the minimum distance from the centroid (see Figure 2.13 A). The curve is then used to identify the head and the tail regions (see Figure 2.13 B). The tail region is assumed to have the highest curvature, while the head region is assumed to be the furthest from the center apart from the tail. The left and right sides are identified as the regions between the head and the

tail regions. The frequency of motion is calculated using a technique similar to the one used in (Nie, Y., et al. 2008). The authors calculate features such as the speed of the center of the silhouette, the total area under the curve that plots the polar coordinates of the contour of the mouse with respect to the angle, the frequency of motion in the head region, left side region, and right side region to identify different behaviors. A lack of motion determines immobility, a large motion speed identifies motion, high motion frequency at the head zone identifies head grooming, and high motion frequency at the body side identifies scratching.

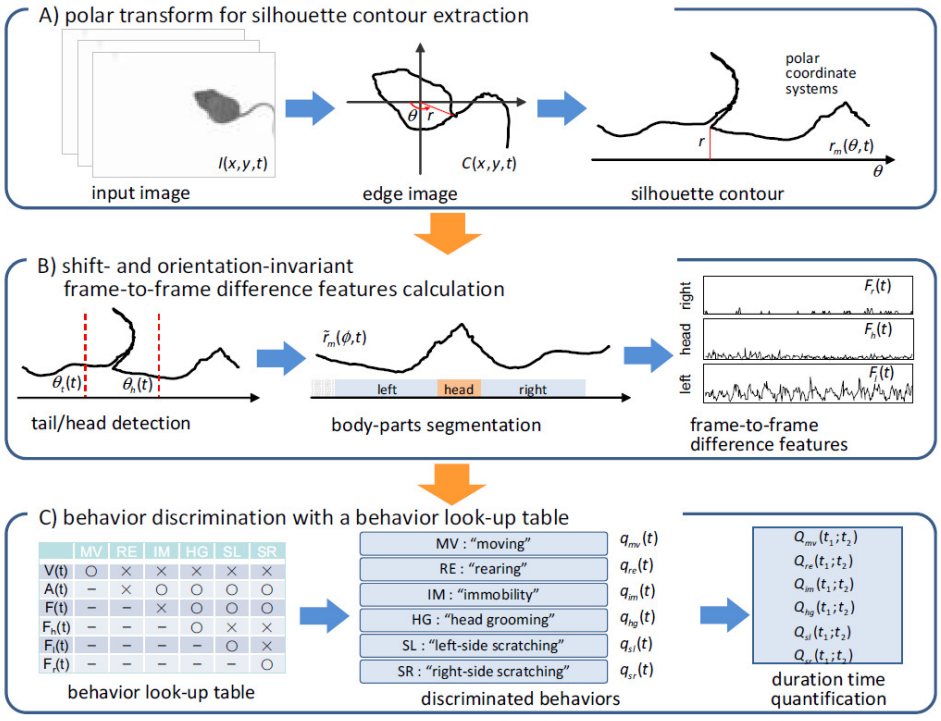


Figure 2.13 The rodent behavior recognition method described in Nie et al. [2011] (© 2011, IEEE)

Dollar et al. (2005) proposed using 3D spatio-temporal features called cuboids to discriminate drinking, eating, exploring, grooming and sleeping in mice. The method first extracts interest points by applying the response function R given in equation 2.19:

$$R = (I * g * h_{ev})^2 + (I * g * h_{od})^2, \tag{2.19}$$

where I is the intensity frame, $g(x, y; \sigma)$ is a 2D Gaussian smoothing kernel, applied along the spatial dimensions, and h_{ev} and h_{od} are a quadrature pair of a 1D Gabor filter (Granlund & Knutsson, 1995) that is applied along the temporal dimension. The response function is designed in such a way that it returns high value whenever the variations in the local image

intensities present periodic frequency components. The cuboids are then extracted at the interest points generated by *R* and they consist of the pixel values along spatio-temporal windows centered on the interest points. Cuboids are then subjected to three transformations: normalization of the pixels, calculation of the brightness gradient and calculation of the windowed optical flow. Features are formed by applying a 2D SIFT (Lowe, 2004) descriptor followed by PCA on the cuboids (Hastie, H., et al., 2001). Finally, each feature is classified by assigning it to the closest prototype feature.

The 3D spatio-temporal feature method does not perform well when applied to mice. The authors explain that the mouse blends in with the bedding of the cage. They also explain that the actions of the mouse were subtle and that there were no easily trackable features on the mouse. This resulted in inaccurate optical flow estimates. Furthermore, this method is limited to dark rodents (Burgos-Artizzu, X. P., et al., 2012).

Jhuang et al. (2010c) also used 3D spatio-temporal features to classify drinking, eating, grooming, hanging, micro-moving, rearing, resting and walking. Their method is limited to dark rodents (Jhuang, H., et al., 2010a). They used three types of features to identify each of the behaviors. The first type consists of the 3D spatio-temporal features first introduced by the authors in (Jhuang, H., et al., 2007). Those features are extracted using a series of 3D filtering and template matching. The authors also used position- and velocity-based features. To compute those features, the authors conducted a background subtraction to extract the target blob. They then drew a bounding box around the blob. Consequently, features such as the position, the aspect ratio of the bounding box, the distance of the animal from a feeder installed in the cage, and the instantaneous acceleration and velocity were computed. A trained SVM classifier was used to classify the features and estimate the behavior of the animal.

HomeCageScan, an industrial solution by CleverSys (Wilks, S.L., et al., 2004; Liang, Y., et al., 2007) uses a background subtraction to isolate a rodent blob in a cage. The background is constructed using several frames and is updated on a regular basis. The rodent blob is then fed to a Hidden Markov Model (HMM) to determine one of the twenty one behaviors that the solution identifies. The HMM uses features such as shape position and movement to classify the blob. However, as mentioned on the website of CleverSys (2013), The system relies on the use of a transparent cage in front of a uniform white background, and of adapted lighting.

The projects implemented by Nie et al. (2008), Nie et al. (2011), Dollar et al. (2005), Jhuang et al. (2010c) and even the industrial product CleverSys all control the environment in a dif-

ferent manner to ensure enough contrast is present between the rodent and the background. The constraints that they impose are not typical of a regular biomedical environment. For this reason, we found it necessary to propose a method to detect rodent behaviors with minimum constraints on the environment of the ongoing experiment.

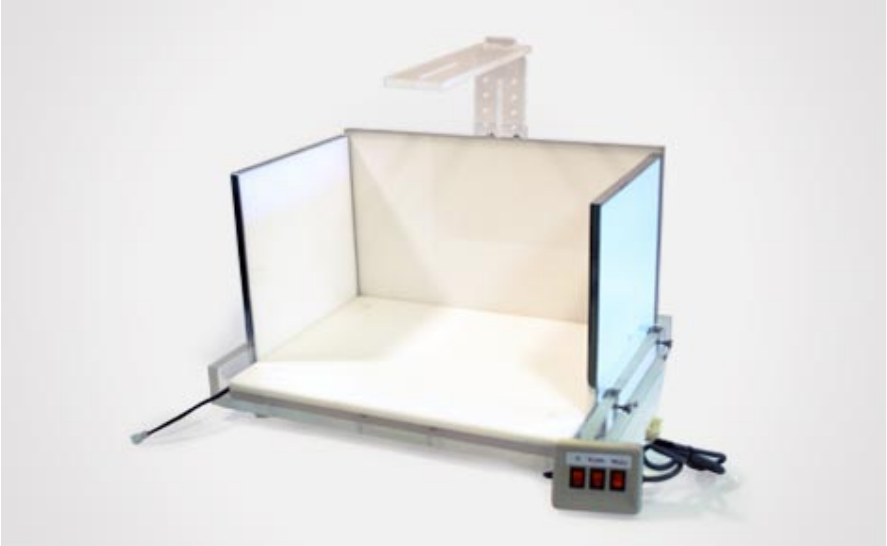


Figure 2.14 Customized environment with specialized lighting used with Home-CageScan for optimized result (CleverSys, 2013)

CHAPTER 3 OVERVIEW OF APPROACHES

This thesis presents the proposed solutions for three related problems concerned with rodent tracking and monitoring using computer vision techniques. These solutions are summarized in three journal articles that are presented in Chapters 4, 5 and 6.

The first article (Chapter 4) tackles the problem of rodent tracking and extraction using computer vision techniques under typical biomedical laboratory settings. Typical biomedical environment have proven to be challenging for computer vision processing. In such environments, the lighting is seldom uniform, cages are stacked on shelves, the cage material is susceptible to a high degree of reflections, and cages are usually covered with bedding that is highly textured and dynamic. Rodents in themselves present major challenges for computer vision techniques because their structure is highly deformable, which makes it difficult to model. Rodents also tend to exhibit erratic and abrupt motion.

In the first article, we proposed a method to track rodents with minimal environmental constraints while taking the outlined challenges into consideration. The method consists of two phases. In the first phase, a sliding window established a coarse track of the rodent. In the second phase, the boundaries of the sliding window are adjusted to fit the contour of the rodent.

The sliding window scans the region around the rodent position in the previous frame. For every position of the sliding window a cost is calculated based on three features:

1. The distance between the HOG calculated using the pixels inside the sliding window at the current position and the window positioned on the target in the previous frame.
2. The distance between the OHI calculated using the pixels inside the sliding window at the current position and the window positioned on the target in the previous frame.
3. The value of absence of motion in the region bounded by the sliding window at the current position.

The current position of the target is assumed to be located at the sliding window position that minimizes the cost.

To extract the rodent from the frame, the method includes a scheme to adjust the boundaries of the sliding window, at the position that minimizes the cost, to fit the target contour. This is

done using an online edge-background subtraction and edglet-based constructed pulses. An online edge-background is a continuously updated background constructed out of the edges that belong to the background in the frame. The online edge-background is used to extract the edges of the rodent in the edge map. Then, a system of pulses is constructed around the rodent that account for the presence of an edge pixel on the horizontal and vertical dimensions. The boundaries of the sliding window are then adjusted to delimit the largest intersection of the projection of the vertical and the horizontal pulses, thus extracting the rodent from the frame.

The second article (Chapter 5) proposes a method to identify and distinguish three behaviors in rodents in a typical biomedical environment. The behaviors are being static, rearing and exploring. The method computes the MHI in each frame then extracts four features out of it. The first feature is the number of pixels that are attributed to the MHI. If that number is zero, then the rodent is assumed to be static. The second and third features are the normalized height of the MHI and the HOG calculated from the MHI's pixels, respectively. These two features are then fed to a trained SVM classifier to determine if the rodent is on two feet or on four feet. If the rodent is determined to be on two feet, then it is assumed to be rearing. Otherwise, the method computes the fourth feature, the displacement of the centroid of the MHI. If the displacement is higher than 10% of the rodent size in the x- or y-dimension, then the rodent is assumed to be exploring.

The third article (Chapter 6) proposes an efficient tracking algorithm based on the particle filter to track rodents and other subjects. The tracking algorithm has the same apparent functionalities and competitive processing quality as the particle filter. However, the tracking algorithm also has a reduced computational complexity as compared to the original particle filter. It dynamically reduces the number of processed particles at each time step, thus allowing a high processing time reduction in software implementations and promoting an efficient hardware implementation. The proposed tracking algorithm also includes a new method for resampling that allows the assignment of the particles at time $t+1$ from the set of particles at time t, while reducing the number of random variable generation and the number of divisions.

CHAPTER 4 CATCHING A RAT BY ITS EDGLETS[2]

Abstract— Computer vision is a non-invasive method for monitoring laboratory animals. In this article, we propose a robust tracking method that is capable of extracting a rodent from a frame under uncontrolled normal laboratory conditions. The method consists of two steps. First, a sliding window combines three features to coarsely track the animal. Then, it uses the edglets of the rodent to adjust the tracked region to the animal's boundary. The method achieves an average tracking error smaller than a representative state-of-the-art method.

4.1 INTRODUCTION

An automated non-intrusive animal monitoring system is of great value in biomedical laboratories. It has the potential to dramatically increase laboratory staff efficiency and productivity by reducing or eliminating the time spent reviewing videos or directly monitoring animals. Consequently, a larger quantity of acquired data can be processed, which can lead to better research results in a shorter time.

Automated video monitoring is done using computer vision systems. However, biomedical conditions impose several challenges to those systems, especially when the animals to monitor are rodents. The challenges can be imposed by the environment or the settings, or by pre-recorded videos that do not present any consideration for automatic video processing. For instance, lighting in biomedical labs is seldom customized for computer vision processing. Cages are usually stacked on shelves, which restricts the position of the camera and the field of view. Cages can also be connected to other devices, and can be made of several materials that give rise to different type of artifacts, like metal that is prone to noisy reflections or transparent glass that is scratched. The rodents can have the same color as their background and the cages usually contain bedding to ensure the comfort of the animal. The bedding is dynamic due to the rodent's motion. More importantly, rat bodies are very deformable. This charac-

---

2. FARAH, R., LANGLOIS, J.M.P. and BILODEAU, G.A. (2013). Catching a Rat by its Edglets. *IEEE Transactions on Image Processing (TIP)*, vol. 22(2), pp. 668-678. (© 2013, IEEE)

teristic makes them hard to model.

The purpose of this article is to extract rodents, from a scene, using a computer vision system under the conditions stated above. The proposed method consists of two steps. The first step combines three weak features to roughly track the target. The second step adjusts the boundaries of the tracker to extract the rodent. One contribution of this paper is the introduction of a new feature which is the overlapped histograms of intensity (*OHI*). We also propose a new segmentation method to extract the target's boundaries using an online edge-background (e-background) subtraction and edglet-based constructed pulses. Edglets are discontinuous pieces of edges. The online e-background is a continuously updated frame constructed out of the accumulation of background edglets. Our method operates in settings that are typical of a biomedical laboratory. In our test conditions, no special cages were used, lighting was unchanged, and the cages were left on their shelves, which constrained video monitoring to a side view. An early version of this work was presented previously (Farah, R., et al. 2011)

The paper is organized as follows: Section 4.2 describes related work on animal tracking. Section 4.3 presents the problem analysis. Section 4.4 details the method that we propose for tracking and extracting animals. Section 4.5 presents the experimental settings and results, and section 4.6 concludes the paper.

4.2 Literature Review

Animal tracking algorithms have been the subject of much research in computer vision because of the available applications and the differences in morphology and behavior between animals and humans. In general, standard human tracking methods cannot be applied directly to animals. Developed methods depend on the applications, and the conditions and limitations of the environmental settings.

For laboratory animals, Pistori et al. (2010) and Goncalve et al. (2007) used a particle filter combined with a k-mean algorithm to track several mice in a cage. The algorithm extracts blobs to calculate the mice's center of mass and the parameters of a bounding ellipse. The algorithm is restricted to processing white mice on a dark background, because the tracking algorithm starts with a segmentation that uses simple color thresholding. The method used by Ishii et al. (2011) suffers from the same restriction. The authors tracked a white mouse on a black background using simple color thresholding for segmentation. They then calculated the center of mass of the foreground to track the rat. Nie et al. (2008) (2009) used the same segmentation principle to track a dark mouse in a transparent container filled with water, or a

mouse in a Plexiglas cage positioned above an IR illuminator. Simple segmentation techniques require a certain level of contrast between the background and the target. This restriction is not always realistic due to environmental settings and requirements of an ongoing biomedical experiment. For instance, the animal's breed is usually imposed by the experiment or its availability and the color of the animal is determined by its breed. Moreover, in some environments, cages are stacked on shelves. In this case, transparent cages are used to enable monitoring by laboratory staff. Furthermore, cage floors are often covered with bedding to ensure the comfort of the animal and avoid stressing it. As a result, the contrast required by a simple color thresholding is seldom available in biomedical environments.

Dollar et al. (2005) and Belongie et al. (2005) used 3D spatio-temporal gradient features to track and detect certain specific behaviors in humans and mice. The method does not use segmentation. The authors mention that their method does not perform well when used on mice. They explain that the number of features formed on mice isn't sufficient due to the properties of their texture. Jhuang et al. (2010a) used 3D spatio-temporal features preceded by a background subtraction process. On their website, the authors mention that their algorithm is restricted to dark mice over a white background (Jhuang, H., et al., 2011). Above all, a color-based background subtraction is not advisable in uncontrolled environment for three reasons. First, the cage may move due to animal motion. Second, if the background is constructed while the rodent is in the cage, the rodent may stay in one place for long durations, and the resulting background will not be reliable as it will contain a phantom shape of the rat at the place where the rat was stationary. Further, because the animal area may encompass a large portion of the image, some area of the background will be seldom visible even if the animal moves. Third, the bedding is also displaced by the animal in the cage. It is impractical to maintain a color-based background that takes into account those displacements.

Dollar et al. (2006) proposed extracting the edges of a target using a multiple feature classifier. The features include gradients, Haar Wavelets, and difference of histograms computed on filtered frames after applying a difference of Gaussian (*DoG*) or a difference of offset Gaussian (*DooG*).

Branson and Belongie (2005) used a particle filter that combines a multiple blob tracker and a contour tracker to track several mice in a cage. The method relies heavily on the animals' contours to fit the tracker. Accordingly, the method requires an edge detector with special characteristics to perform. The Berkley Segmentation Engine (Issard & MacCormick, 2001) was

chosen, using 12,000 annotated images to train the edge detector. In addition, it took the detector over five minutes to process one image.

Many commercial solutions exist for rodent tracking. We are aware of only two that provide a computer vision solution. The first solution, by Noldus, provides a tracking software named EthoVision XT (Noldus, 2011a). Details of the video tracking method are not available. According to a demo video (Noldus, 2011a), a template should be defined for each experiment. The template indicates the species of the animal to be tracked, the dimensions and the edges of the cage. Manual adjustment is also required to align the template and the cage. Specific cages are provided by the company. A combination of a visible camera and an infrared camera is required to reduce the sensitivity to fur color variations and to reduce problems caused by light reflections (Noldus, 2011b). The second solution, by CleverSys (Wilks, S.L., et al., 2004; Liang, Y., et al., 2007) uses a color-based background subtraction technique to extract the rodent as foreground. It then calculates the rodent's center of mass for tracking. For optimal results, the software needs a specialized system that provides adapted lightning and a uniform white background (CleverSys, 2013).

After considering the previous analysis, we observed that successful target extraction should be done with a method that does not rely solely on color-based background extraction, a single feature, or restrictive and tedious pre-training. Thus, we propose a track-and-refine edglet-based method that does not impose any restriction on actual environment settings as long as the cage is transparent.

4.3 Problem Analysis

We conducted an analysis based on several features and schemes to determine which is more suitable for target extraction.

When considering the texture, and in particular the gradients, we observe that it is difficult to model our target because its texture is extremely variable. In fact the rat has a very deformable body on which the fur changes in appearance as the rat is moving. For instance, when applying, KLT (Shi & Tomasi, 1994) to our data set, the feature points moved out of the target after a certain number of frames even though the tracker was forced to initialize on the target (see Figure 4.1). This behavior is consistent with (Dollar, P., et al., 2005) and (Belongie, S., et al., 2005) as gradients were also used in those papers. Another reason why KLT failed is that the target has less texture with respect to its environment in some cases.

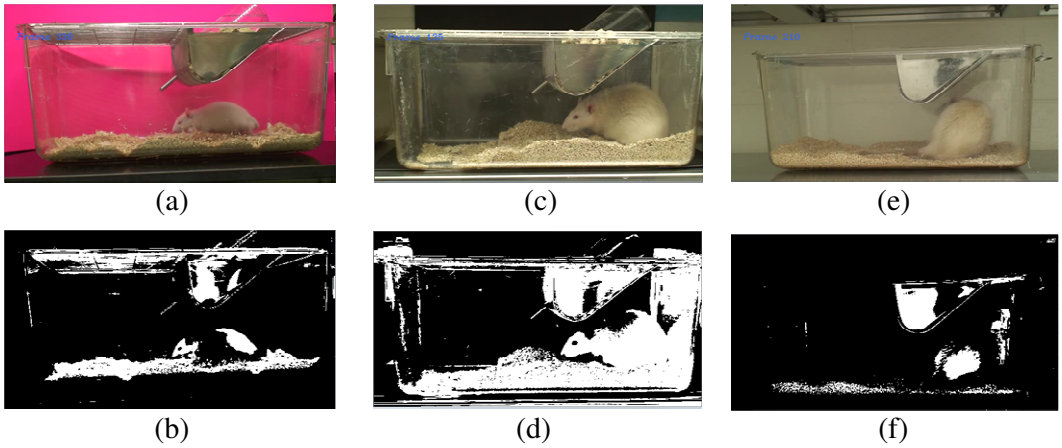


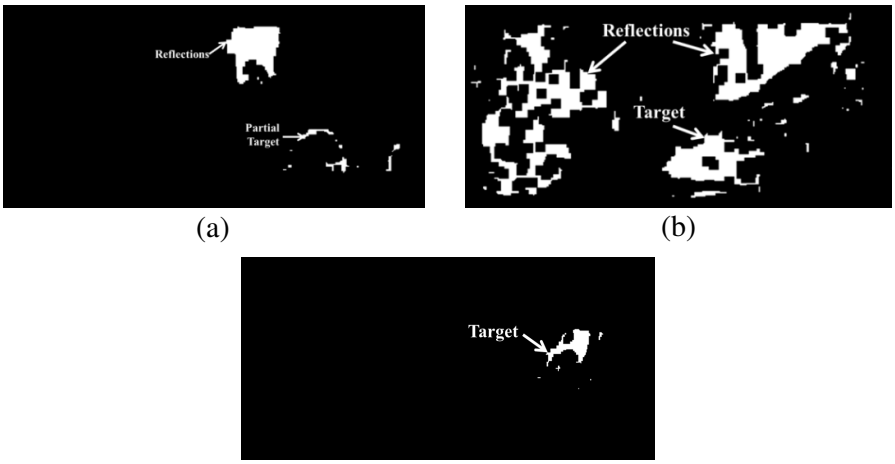
Figure 4.1 Target histogram projection. (a,b) a change in the target color distribution cuts the target in two. (c,d) The target and the background share common color zones that give a larger apparent size to the target. (e,f) A change in color distribution of the target gives it a smaller apparent size.

When considering color, the target's appearance and color distribution are extremely dynamic. The rats are often multicolored and share colors with the background. An analysis of the histogram projection of a Mean Shift algorithm (Comaniciu & Meer, 2002) reveals that the target is divided into several color bands. Figure 4.2 shows that the background and the target have common colors. In fact, a Camshift algorithm (Bradski, 1998), which is a tracking algorithm based on the Mean Shift segmentation, loses tracking after a few frames. These challenges are common to most color segmentation methods such as (Nie, Y., et al., 2011), (Nie, Y., et al., 2008), (Nie, Y., et al., 2008) and (Wilks, S.L., et al., 2004).

Figure 4.2 Point Features for the KLT tracker at (a) frame 1: eventhough the max number of point features was set to 500, KLT initialized with 20 point features on the target, (b) frame 1593: only seven point features remain on the target, (c) frame 2531: only two feature points remain on the target.

Another concern arises in the case that the target stays immobile for a long time. This would pose an important challenge to a method similar to the Gaussian Mixture Model (GMM) (Stauffer & Grimson, 1999). GMM is a robust color based segmentation method. Its particularity is that it takes into consideration the background information. In fact, the background is

represented by several models, in this case three, that are regularly updated. However, GMM's greatest weakness is that when the target is stationary for a long time it tends to blend with the background (See Figure 4.3 (c)).



(a) (b)

(c)

Figure 4.3 Foreground segmentation using GMM

We also investigated contour features. Contour features are easily distracted by noisy edges. This is also the case in an active contour model (Hao & Drew, 2002) that typically use contour features. Furthermore, it is challenging to control their energy function when the target is very deformable. In (Vaswani, N., et al., 2010), the authors used a particle filter based on active contour method. To address the previous shortcomings, the authors used both edge information and color information to adjust their contours and they relaxed their energy function. When used on our dataset, this method still failed because both the contour and color distribution of the target were extremely dynamic.

Assuming that the target is the only mobile entity in the frame, motion would form a strong indicator on the position of the target. However, this is not always the case. First, the target may remain immobile for certain duration. Second, the dynamic bedding and the reflections of objects moving in front of the cages give the impression of motion. Their effect is most significant when the target is stationary as the position of the target could be mistaken with the position of the reflections or the moving bedding.

According to this analysis, we conclude that color, gradients and motion based features are weak features by themselves. This justifies the conclusion reached in the previous section. Neither gradient based features nor color information appear to be sufficient by themselves to extract a target that has such dynamic shape, texture and color distribution from a dynamic

background that shares some of the color distribution of the target.

4.4 Target Extraction

The proposed method uses a sliding window approach to coarsely localize the target. It then adapts the tracked region boundaries to fit the contour of the target using the target's edglets and an e-background subtraction. Adaptation is necessary to account for changes in the animal's pose, scale and deformation.

4.4.1 Coarse Animal Localization

The sliding window scans the vicinity of the rodent's previous position to estimate the current position. The window's dimensions are kept fixed ($N \times M$) for all frames due to several reasons. First, the rodent's body is very deformable and may change shape quickly. Accordingly, it is very difficult to predict and adapt the window size automatically and reliably. Second, the size of the window affects the speed of computation. As a result, if we vary the size of the window to test several size and scale hypotheses, the processed frame rate would be negatively affected. In addition, the window size may become unstable and grow or shrink indefinitely due to noisy structures that can be mistakenly associated with the target.

The system is initialized by manually drawing the first window around the target in the first frame. The target posture and position are irrelevant given that in the next phase the boundaries of the window will be adjusted. A large tracking window also leads to an increase in processing time. Consequently, if the target occupies a large area in the first frame, it is better to select a smaller window on the body of the target, taking into consideration that the window should be large enough to include as much information about the target as possible. The window content should be representative of the target color distribution and texture and avoid background zones.

For each displacement of the sliding window, the bounded region is considered as a candidate. The target is chosen as the candidate region that minimizes a cost function (S_f). The cost function S_f is based on a composite strong feature that combines three weak features: the histograms of oriented gradients (*HOG*), the overlapped histograms of intensity (*OHI*), and the absence of motion A_m.

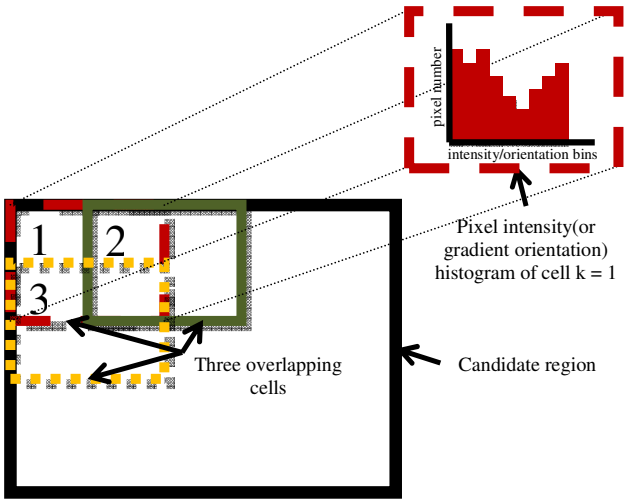


Figure 4.4 HOG and OHI calculation

Histograms of oriented gradients (HOG)

HOG was chosen to account for texture. *HOG* is designed to be mostly invariant to the target's geometric transformations and changes in illumination. The *HOG* feature is based on the algorithm described by Dallal et al. (2005) (2006). *HOG* is calculated as follows:

1) The candidate region is divided into $n \times m$ overlapping cells and k will refer to the index of one of the cells.

2) The gradient orientation histogram h_{HOG} is computed for each cell.

$$h_{HOG}^{k}(r_a) = \sum_{p=1}^{n_a} i_p, \tag{4.1}$$

where r_a is an orientation interval, n_a the number of gradients that have an orientation included in r_a, and i_p the magnitude of each gradient.

3) The *HOG* feature is, then, constructed as an $nm \times a$ matrix where

$$\mathrm{HOG(k)} = \frac{\mathrm{h_{HOG}^{k}}}{\|\mathrm{HOG}\|}, \tag{4.2}$$

where h_{HOG}^{k} is the histogram of the k^{th}cell and $\|HOG\|$ is the max norm of the $nm \times a$ *HOG* matrix.

Overlapped histograms of intensity (OHI)

The histogram of intensity (*HoI*) is the classical method to model a region of interest's intensity distribution. Comaniciu et al. (2003) used *HoI* to detect and track people in a scene. Yin

et al. (2003) combined *HoI* with *HOG*, the motion history image (*MHI*) (Yin & Collins, 2006), the saliency likelihood map (Hou & Zhang 2007) and the template likelihood map (Yin, Z., et al., 2008) to detect objects in a scene. Histograms of intensity are commonly used because they are robust to change in scale and rotation. However, *HoI* fails to capture local intensity distributions. Inspired by *HOG's* structure, we propose the overlapped histograms of intensities (*OHI*s) as a compromise between *HoI* and the fine granularity provided by individual pixels, because it is calculated on small cells. *OHI* calculation is similar to *HOG's*.

1) The candidate region is divided into $n \times m$ overlapping cells.

2) An intensity histogram of b bins is calculated for each cell as follows:

$$h^k(r_b) = n_b, \tag{4.3}$$

where r_b is one of the calculated intensity intervals, n_b is the number of pixels in the frame which intensities are included in r_b.

3) We constructed the *OHI* feature matrix as a $nm \times b$ matrix where

$$OHI(k) = \frac{h^k}{\|OHI\|}, \tag{4.4}$$

h^k is the histogram of the k^{th}cell and $\|OHI\|$ is the norm for the $nm \times b$ *OHI* matrix. *HOG* and *OHI* are illustrated in Figure 4.4.

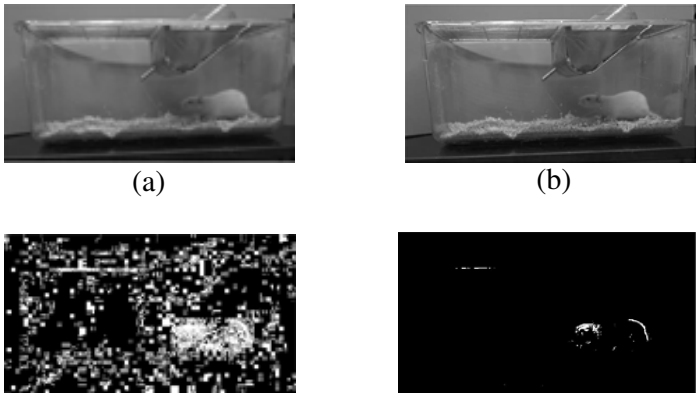
Absence of motion A_m

The target should be the only mobile entity in the cage. Accordingly, any detected motion is a strong indicator to the position of the target, hence the utility of using the absence of motion (A_m) in our cost function.

$$D = \begin{cases} 1 & if\ |F_t - F_{t+1}| > \in_1 \\ 0 & otherwise \end{cases}, \tag{4.5}$$

$$A_m = (M \times N) - \sum_{i=1}^{M \times N} D, \ , \tag{4.6}$$

where F_t and F_{t+1} are two consecutive grayscale frames, and $\in_1$ a given threshold Figure 4.5 illustrates A_m calculation.



(a) (b) (c) (d)

Figure 4.5 $\mathbf{A_m}$ calculation. (a) Frame $\mathbf{t}$, (b) Frame $\mathbf{t+1}$,(c) $|(\mathbf{Frame\ t}) - (\mathbf{Frame\ t+1})|$, (d) The frame after thresholding (© 2011, IEEE)

Cost function

The cost function (S_f) is calculated as follows

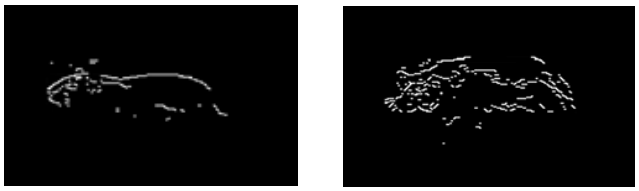
$$S_{f(w_i)} = \alpha_1 D_{HOG(w_i)} + \alpha_2 D_{OHI(w_i)} + \alpha_3 A_{m(w_i)}, \tag{4.7}$$

where w_i is the i^{th} candidate window, D_{HOG} is the distance between the *HOG* feature matrices of the target window at time t and the candidate window at time $t+1$, D_{OHI} is the distance between their *OHI* feature matrices. D_{HOG} and D_{OHI} are calculated using the Euclidian distance. The Euclidean distance is used because it satisfies the requirements of the method in addition to being simple to implement. The alphas are weights that are attributed to each component of the cost function.

4.4.2 Boundary Refinement

The sliding window tracker approximates the location of the target, but its dimensions and position should be adapted to extract the target from the frame. We found that the edge map is useful to undertake the boundary refinement of the window because we can isolate many of the target's edglets. The edge map is also advantageous with respect to the color map because it is insensitive to the changes in the color distribution of the target, and because it allows for a simple way to isolate the highly textured regions. In addition, the probability of the back-

ground and target sharing similar color zones is greater than the probability of the background and the target sharing edges. This is due to the significantly smaller area that an edge covers with respect to a color zone. Consequently, in the edge map it is less likely to assign target parts to the background than in the color map.



(a) (b)

Figure 4.6 Edglets (a) an edge map in a high contrast video (b) an edge map in low contrast video

Edglets

When computing the edge map, the resulting contour of the rat is not continuous. It is composed of short groups of edges that we refer to as edglets. The discontinuity of the contour is due to contrast imperfection between the rat and the background. Figure 4.6 shows examples of edglets. The edglets are longer and fewer in number when the contrast is more pronounced.

The edge map contains a large number of background edglets. These edglets will cause distraction to the refinement process and result in incorrect target dimensions. To suppress these edglets, we propose a new method for background edge subtraction. Similarly to color-based background subtraction, we aim at removing from the foreground the edges (and thus edglets) that belong to the background.

Online e-background

The e-background consists of the background edglets. In fact, an e-background could be constructed in the same manner as a color-based background. Nevertheless, instead of using the repetition of the same pixel's intensity to include it in the background, we consider its edglet occurrence repetition to include it in the background.

Given eB_{t-1} the online e-background calculated at time $t-1$, eC_t the edgelet map calculated at time t, and eT_{t} the edglets in eC_t that are included in the target's bounding box, the online e-background at time t is calculated as follows:

$$eB_t = \alpha \times (eC_t - eT_t) + (1-\alpha) \times eB_{t-1}, \tag{4.8}$$

For this operation, the target coordinates are taken as the ones calculated at $t-1$. eC_t is calculated using Canny's algorithm (Canny, 1986) applied on the gray scale frame. The OpenCV libraries were used to compute gray scale transformation and the edge map using Canny's algorithm.

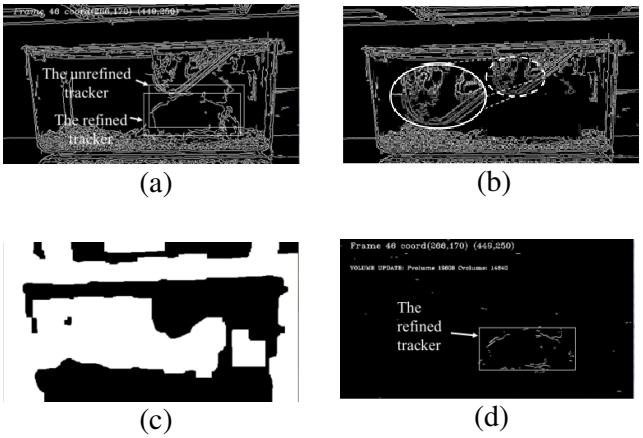


Figure 4.7 Online e-background subtraction. (a) an edge map. (b) the constructed e-background. (c) the map of edge high density. (d) the foreground edglet map.

The e-background and the result of the e-background subtraction are illustrated in Figure 4.7. Notice in Figure 4.7(b), the edglets that are constant in the e-background are represented with higher values (lighter colors) due to accumulation.

However, because the edglets in highly textured regions with weaker magnitude have a tendency to shift position because of noise from one video frame to the next, considering only the (x,y) coordinates to detect foreground edglets does not give satisfying results. Thus, we must account for the slight edglet shifts that may occur between frames in high edglet density background regions. For this purpose, we use an operator, D_{He}, to evaluate the edglets density in the e-background. In other words, the edglets density is calculated by dividing the edge map into a grid of $\eta \times \eta$ pixel squares. Given $Ne(l)$ as the number of edglet pixels in the l^{th} square of the grid, the edglets density, in the l^{th} square is calculated as:

$$D_{He}(l) = \begin{cases} 0 & if\ N_e(l) > \in_2 \\ 1 & otherwise \end{cases}, \tag{4.9}$$

Figure 4.7(b and c) shows, respectively, a frame edge representation, and the result of calculating D_{He}.

Finally, the target edglets eT are constructed from all the pixels that are excluded from eB

and D_{He}

$$eT(i_e) = \begin{cases} 1 & if\ (\overline{eB_{t-1}(i_e)} \times eC_t(i_e)) > \in_3\ and\ D_{He}(i_e) = 1 \\ 0 & otherwise \end{cases}, \quad (4.10)$$

where i_e is a pixel in the edglets map, $\overline{eB_{t-1}(i_e)}$ is the complement of the background and $D_{He}(i_e)$ is edglet density of the square to which that pixel belongs.

Final Localization of the Animal

Using the foreground edglets, we now reconstruct the regions corresponding to the animal. To do that, the vicinity of the tracking window is scanned to produce pulse graphs in the edge map (Figure 4.8(b)). The horizontal pulse graph P_h is constructed by scanning the region horizontally.

$$P_h(x) = \begin{cases} 1 & if\ \sum_{y=y1}^{y2} F(x,y) > 0 \\ 0 & otherwise \end{cases}, \quad (4.11)$$

where $y1$ and $y2$ are the upper and lower limits of the scanned region and $F(x,y)$ is the intensity of the pixel at (x,y) in the edge map.

Similarly, to produce the vertical pulse graph P_v, the region is scanned vertically and the pulses are produced.

$$P_v(y) = \begin{cases} 1 & if\ \sum_{x=x1}^{x2} F(x,y) > 0 \\ 0 & otherwise \end{cases}, \quad (4.12)$$

where $x1$ and $x2$ are the left and right limits of the scanned region and $F(x,y)$ is the intensity of the pixel at (x,y) in the edge map.

We assume that the rat's edglets are close to each other; consequently their corresponding pulses are close to each other as well. Those pulses are merged to assemble the complete target entity coverage. This procedure is added because the contour of the target is seldom completely continuous. Its objective is to merge its constituent edglets.

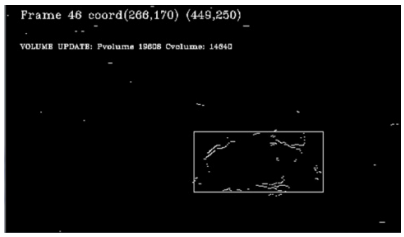
The merge process of positive pulses, for the horizontal pulse series (P_{h_M}) is described in (4.13). Short zero pulses are detected and converted to positive pulses:

$$P_{h_M}(x) = \begin{cases} 1 & if\ P_h(x) = 1 \\ 1 & if\ d < \epsilon_4\ and\ P_h(x, x+d) = 0, \\ 0 & otherwise \end{cases} \tag{4.13}$$

where d is the length of a zero pulse and $P_h(x, x+d)$ is a pulse that has a magnitude of zero and extends between the coordinates x and $x+d$. In other words, a point on the pulse graph is assigned a magnitude of 1 if it belongs to a pulse of magnitude 1 or if it belongs to a pulse of magnitude 0 which has a width $d < \epsilon_4$. The same procedure is applied to the vertical pulse series.

(a)

(b)



(c)

Figure 4.8 Boundary Refinement. (a) an edge map after e-background subtraction. (b) The calculated pulses. (c) The tracking window after refinement.

Afterwards, the resulting pulses are projected on the frame, and the region R_t^{max} that maximizes the intersection of two of those projections is chosen to set the boundaries of the tracker according to the following condition

$$T_t = \begin{cases} R_{max} & if\ R_{max} > \beta T_{t-1} \\ T_{t-1} & otherwise \end{cases}, \tag{4.14}$$

where

$$R_{max} = max(Proj(P_{h_M}(i)) \cap Proj(P_{v_M}(j))), \tag{4.15}$$

$Proj\left(P_{h_M}(i)\right)$ and $Proj\left(P_{v_M}(j)\right)$ are the projections of two piecewise constant functions that belong to P_{h_M} and P_{v_M}, respectively. T_t and T_{t-1} are the bounding boxes at time t and

$t-1$ respectively .

We also assume that the target cannot have a drastic change of area between two consecutive frames. This is why an area change constraint is applied in (4.14).

4.5 Experimental Results

To test the proposed methodology, videos of rats in cages were recorded at the research center in Sainte-Justine Children Hospital. The camera set up and data acquisition did not disturb the ongoing experiments, or the actual environment conditions. The cages were set on shelves and illumination was provided by florescent lamps from the ceiling. The cages had the same dimensions, were transparent, and in some of the cases, a pink or a black cardboard was placed behind the cage. The introduction of the colored cardboard is for the purpose of testing a variety of backgrounds. The information about the experimental settings is summarized in Table 4.1. For video 3, no cardboard was added while for video 1 and video 2, a pink cardboard and a black cardboard were added, respectively (Figure 4.9). The rats were white and of different size classes. The real dimensions of the rat were not recorded. We measured the maximum pixel length recorded for each rat and normalized it with the pixel width of the frame according to (4.16)

$$Relative\ rat\ size = \frac{\max(rat\ pixel\ length)}{(cagewidth_{min}+cagewidth_{max})}, \tag{4.16}$$

Although this is not an accurate method to measure the dimensions of the target, still, it illustrates the size classes of the targets.

Table 4.1 Video information

	Video 1	**Video 2**	**Video 3**
Width	560		
Height	304		
Frame Number	8235	15279	8835
Frame rate	25 frames/second		
Animal color	white	white	white
Background color	pink	black	environment wall (white)
Relative rat size	0.58	0.95	0.82

Table 4.2 summarizes experimental parameters. Even though the parameters are determined empirically, they appear not to be dependent on the size of the rat, nor the colors involved,

given the extent of the experiments. In fact, the same parameters gave the optimal results when used for the three videos. The parameters were chosen after extensive testing on Video 1. However, different variations did not improve the results on the other two videos. Essentially, the parameters were set to achieve a reasonable balance and compromise. For instance, the values obtained for A_m (the absence of motion) were two orders of magnitude larger than the values obtained for D_{HOG} or D_{OHI} (the distances calculated for the histograms of motion and the overlapped histograms of intensity). Accordingly, the values of α_1, α_2 and α_3 were initially set to balance the contributions of the three features. The size of the edge map grid cells ($\eta \times \eta$ pixels squares) was chosen as a granularity compromise. The cells should be small enough to insure a fine granularity and large enough to contain sufficient information. The corresponding threshold ϵ_2 was obtained by training. Samples were collected on high edge density regions and their average was computed. ϵ_1 is set as a compromise between removing as many noise pixels as possible while preserving the motion pixels. Similarly, ϵ_3 is set as a compromise between removing as many noise pixels as possible while preserving foreground pixels. ϵ_4 is set after observation of the width that separates edglets on the target contour. ϵ_4 should not be too high so that noise edglets are combined with the target. We do not expect the target area to change significantly between two consecutive frames. Therefore, we restricted the change of the area by the factor β.

For *HOG* and *OHI* calculations, we used the values suggested in (Dallal & Triggs, 2005) for the cell's dimensions and the number of histogram bins.

Table 4.2 Experimental parameters

parameters	**Equation involved**	**values**
$(\boldsymbol{\alpha_1, \alpha_2, \alpha_3})$	Parameters used in the cost function calculation (7)	(1,1, 0.01)
$\boldsymbol{\alpha}$	e-background update factor (8)	0.4
$\boldsymbol{\eta \times \eta}$	edge map grid dimensions for edge high density calculation	15×15
$\boldsymbol{\in_1}$	Threshold used in motion extraction (5)	50
$\boldsymbol{\in_2}$	Threshold used in edge high density calculation (9)	5
$\boldsymbol{\in_3}$	Threshold used in e-background subtraction (10)	0.5
$\boldsymbol{\in_4}$	Maximum zero pulse width (13)	20
$\boldsymbol{\beta}$	Parameter used for area adaptation restriction (14)	1.7

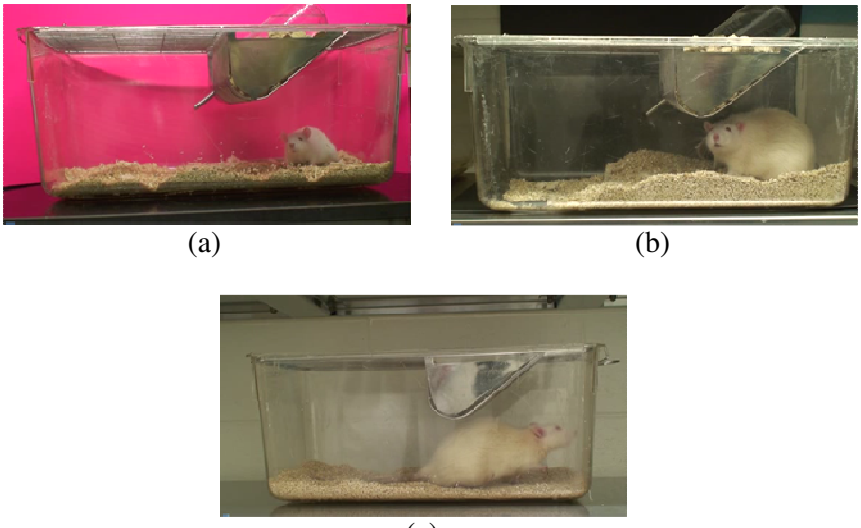
4.5.1 Feature Validation

To evaluate and to validate the three features (*HOG*, *OHI*, A_m) in (7), we considered two strategies. The first one is to isolate each feature and to measure its contribution to the tracking result. The second one is to measure the effects of its absence on the tracking result. For both strategies, only the tracking part of the algorithm was considered (section 4.4.1).

The three videos were tested. For each execution, 200 frames were selected randomly for evaluation. The ground truth was selected as the center of the animal. We performed a manual segmentation to extract the target in each frame. Then, the center of the animal was calculated as the center of its bounding box. The same frames were used in each video for all combinations tested. The calculated error, err_{center} (4.17) is the error between the ground truth center position and the tracker's center position normalized by the tracker's window size.

$$err_{center} = \sqrt{\left(\frac{x_{gt}-x_t}{tracker\ width}\right)^2 + \left(\frac{y_{gt}-y_t}{tracker\ height}\right)^2} \times 100, \qquad (4.17)$$

where, (x_t, y_t) are the coordinates of the tracker center, and (x_{gt}, y_{gt}) are the coordinates ground truth center.



(a) (b)

(c)

Figure 4.9 Snapshots from (a) video 1, (b) video 2, and (c) video 3

The mean and standard deviation (std) of the error, in pixels, are calculated and displayed in Table 4.3. For video 3, the combination of "*OHI* and A_m" gave the best result. In fact, it has a minor advantage on the combination of the three features. However, for video 1, the combination of "*OHI* and A_m" did not match the performance of the combination of the three features

and was even outperformed by the "*HOG* and *OHI*" combination. For video 2, the combination "*HOG* and A_m" gave the best results but not far from the result that corresponds to the three features. Finally, no single feature or combination of two features was dominant in general. The combination of the three features is the most stable and is necessary to ensure good tracking in most cases. It is important to note that this is not the final error. It is the tracking error which is further reduced in the boundary refinement phase.

4.5.2 Comparison with the state-of-the-art

To compare our complete method with the state-of-the-art, we have selected the method described in (Vaswani, N., et al., 2010), which is similar to a method previously applied to rodents (Branson & Belongie 2005), and for which source code was provided. The algorithm described in (Vaswani, N., et al., 2010) is meant to extract the exact contour of the target while the proposed algorithm draws a bounding box around the target. To compare both algorithms, we considered the bounding box around the contour calculated using (Vaswani, N., et al., 2010).

To objectively evaluate the quality of the complete algorithm, the ground truth set was built by drawing a manual bounding box around the rodent at each of 200 randomly selected frames.

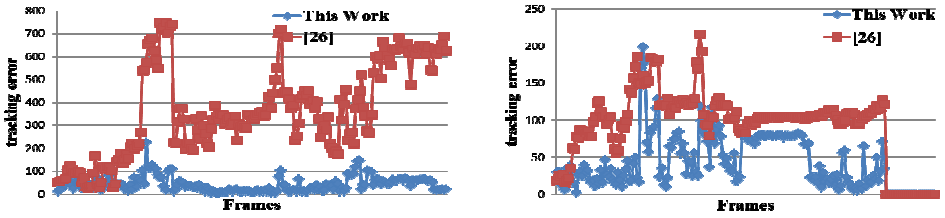
Table 4.3 Feature evaluation(%)

Features	**Video 1**		**Video 2**		**Video 3**	
	Mean error	**std**	**Mean error**	**std**	**Mean error**	**std**
HOG	107.05	50.62	66.30	26.46	62.27	37.89
OHI	49.32	49.32	54.52	22.57	63.59	44.16
A_m	61.68	20.62	81.87	19.18	58.93	23.14
HOG* and *OHI	41.43	28.32	66.78	21.88	58.48	39.07
***HOG* and A_m**	43.97	43.97	41.78	17.90	49.80	44.98
***OHI* and A_m**	43.89	43.89	42.67	15.88	18.98	10.21
All features	36.90	29.12	42.09	14.55	22.90	11.044

The results of the percentage coverage area error, for the algorithms, are shown in Figure 4.10 while the means of the results are summarized in Table 4.4. The same frames were used to evaluate both algorithms. The mean coverage area is calculated as follows:

$$err_{area} = \frac{(Area_T \cup Area_{GT}) - 2\times(Area_T \cap Area_{GT})}{Area_{GT}} \times 100, \qquad (4.18)$$

Where $Area_T$ is the area covered by the tracker and $Area_{GT}$ is the area covered by the ground truth.



(a) (b)

Figure 4-10 Tracking results from [Vaswani, N., et al., 2010] and this work. (a) video 1. (b) video 2

Our algorithm's clear advantage over the algorithm described in (Vaswani, N., et al., 2010) is mainly due to the first step (the tracker). The tracker does not only rely on colors and gradients alone to track the animal, it also exploits motion information. Thus, the tracker steers the boundary refinement step. This keeps the boundary refinement module from being trapped by noise edges. In comparison, the tracker in (Vaswani, N., et al., 2010) cannot recover if it makes tracking errors when distracted by nearby noise. Another reason for the difference in performance is the e-background subtraction which reduces the distraction to the algorithm caused by background edges.

Table 4.4 Percentage coverage area Error

	Video 1	**Video 2**	**Video 3**
(Vaswani, N., et al., 2010)	366.14	Failed	106.97
This work	42.98	46.61	47.80

When considering the performance of our algorithm for the three videos, the advantage in video 1 is due to the pink background. The pink background ensures better contrast between the target and the background and produced fewer reflections. In video 2, the black background produced a mirror effect that highly increased the creation of reflections. The reflections are the main source of noise. This noise is difficult to suppress even with e-background subtraction. In video 3, the contrast between the background and the target is low. Therefore, the edglets created are shorter and more discontinuous. This increases the probability of giving a smaller target apparent size when two consequent edglets are more than 20 pixels apart.

Figure 4.11 shows the calculated error for every ground-truth frame for the proposed algorithm. The frames are sorted chronologically. Figure 4.11 shows some cases where the error

goes above the average. This error is mainly due to the discontinuity in the rat's edglets, which splits the rat in several parts during boundary refinement. Figure 4.12(c) illustrates this case. The edglet on the upper boundary of the rat is split in two. These two parts are separated by more than 20 pixels and are not joined together during the refinement stage. Consequently, the rat will be represented by two separate pulses in the horizontal pulse graph, and only one of them will be chosen to represent the width of the rat.

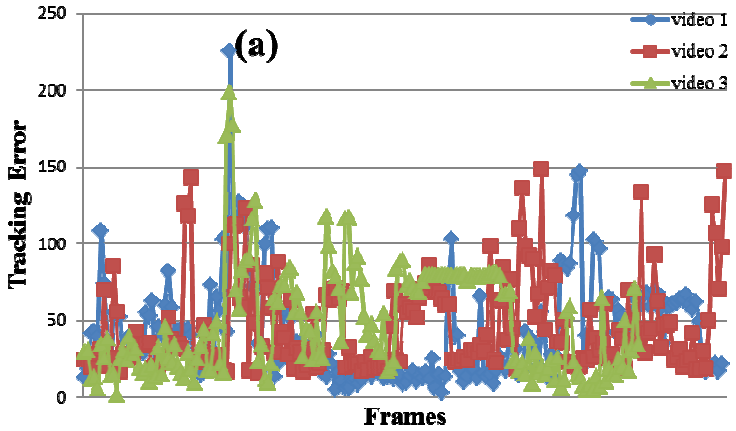


Figure 4.11 The calculated error for the randomly selected frames.

Shadows are another cause of error. Shadows do not belong to the background so they cannot be suppressed while doing the e-background subtraction. The effect of shadows is illustrated in Figure 4.12(b). In that situation, the shadow produces a larger apparent target size. Shadow detection algorithms could be used to minimize their effects on the tracking process.

Noise edglets that are too close to the rat are another source of error. For example, in Figure 4.12(a), some noise edglets, that were not removed by the e-background subtraction are close to the upper boundary of the rat. These edglets were considered as part of the rat and resulted in a larger apparent height.

Despite these shortcomings, the method's performance is consistent under all the conditions tested, as shown in Table 4.4. The method also proved to be efficient in uncontrolled hard settings. In video 3, even though both the rat and the background were white, the method still performed with a mean error less than 48%. Another advantage of the proposed method is that even when the tracker is subject to error, the error does not have a permanent effect and the algorithm can recover at any time. In Figure 4.11, we see that even though the tracker had a big error at point (a), it was able to recover later and reduce the error to zero. Figure 4.13

shows extraction results for the three videos.

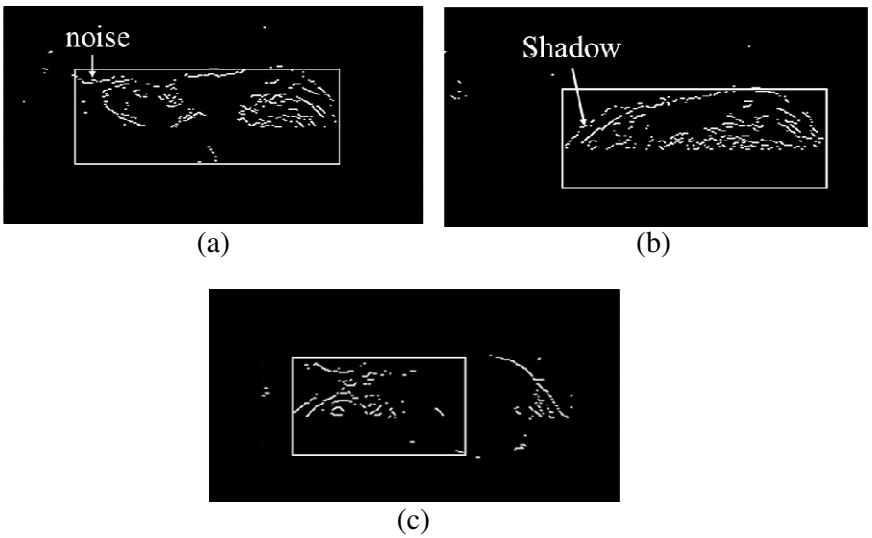


Figure 4.12 source of error in boundary refinement. (a) error cause by nearby noise in the e-background. (b) error caused by shadows. (c) error caused by far target edglets.

Table 4.5 shows results for initializing the tracker with different postures and positions of the target. The tracker was initialized at different frames for the same video, and the same frames were used to calculate the error. Figure 4.14 shows the different postures and positions. The results show that the tracker is insensitive to the initialization window.

Table 4.5 Percentage position error for different initialisation (Video 1)

Video 1	**Frame 1**	**Frame 180**	**Frame 530**
	42.98	66.01	43.75
Video 2	**Frame 1**	**Frame 1250**	
	46.61	46.81	
Video 3	**Frame 1**	**Frame 180**	**Frame 230**
	47.80	43.19	60.29

4.6 Conclusion

In this paper, we proposed a robust method that tracks a rodent in a cage under uncontrolled conditions. The method is formed of two steps. In the first step, three weak features are used to coarsely track the target using a sliding window approach. Step two considers the frame edglets maps to adjust the limits of the tracker to the boundaries of the target. The method uses e-background subtraction to extract the target edglets. Pulses are constructed using the remaining edglets and are used to adjust the tracking window. The method was tested under

representative biomedical environment conditions. The method's performance is consistent when applied to three videos exhibiting different backgrounds and rat sizes.

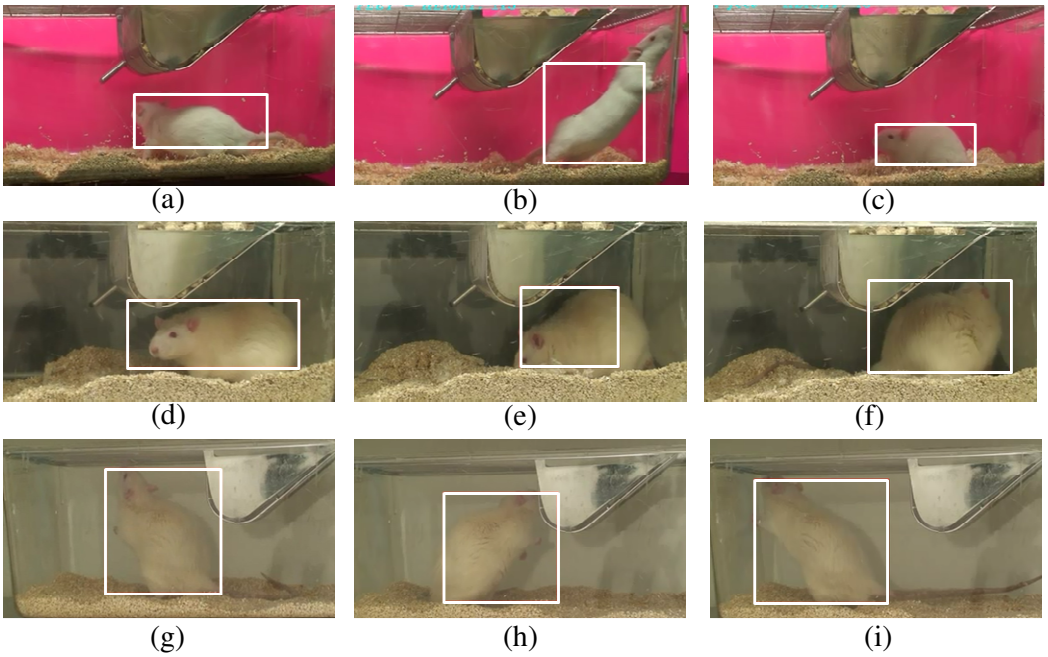


Figure 4.13 Tracking results. (a,b,c) video 1. (a) the extraction error is caused by nearby noise to the rat. (b) the error is caused by groups of edglets that are too far away. (c) successful target extraction. (d,e,f) video 2. (d) the shadow causes a larger target apparent size. (e) error caused by two widely separated pulses. (f) successful target extraction. (g, h, i) video 3. (g, h) error caused by nearby noise. (h) successful target extraction.

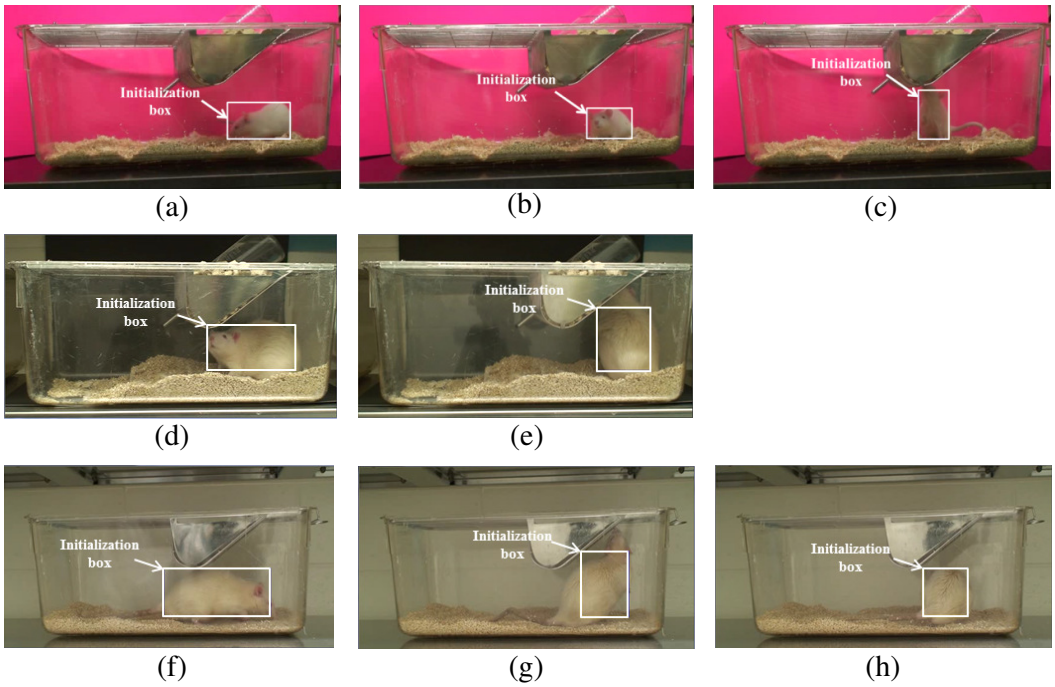


Figure 4.14 Diffirent initialization instances: (a) Video1 - Frame 1, (b) Video 1 - Frame 180, (c) Video 1 - Frame 530, (d) Video 2 – Frame 1, (e) Video 2 – Frame 1250, (f) Video 3 – Frame 1, (g) Video 3 – Frame 180, (h), Video 3 – Frame 530.

CHAPTER 5 COMPUTING A RODENT'S DIARY[3]

Abstract — Rodent monitoring in biomedical laboratories is a time consuming and tedious task. Several automatic solutions that rely on different types of sensors have been proposed. Computer vision provides the most universal and less intrusive solution. In this article we propose a new method to detect and classify three behaviors in rodents: Exploring, Rearing, and Static. The method uses Motion History Images and a Multiple Classifier System to detect the three behaviors under typical laboratory conditions. It is independent of the color of the rodent and of the background. The method performs equally well on short and long video sequences, achieving a success rate of 87%.

5.1 Introduction

Animal models are powerful and proven approaches to perform research in the health sciences. In such a model, animals are used to test drugs to verify their applicability for human treatment or to study diseases, such as epilepsy. Rodents are particularly useful animals because of their physiology's similarity to that of humans. Rodents make up more than 90 percent of animals used for medical research (CHR, 2013). In a typical experiment, a rodent is monitored over a period of hours, days or weeks for specific states and behaviors. Animal monitoring is time consuming and tedious. Experimenters must devote a significant portion of time monitoring and classifying rodent behavior. Biomedical laboratories would benefit from automated, quantitative monitoring devices. These devices would allow for longer observation time leading to more accurate results and conclusions. Several solutions that rely on traditional sensors such as pressure, vibration and infrared beams have been in use. These are reported in (Ishii, H., et al., 2007) and (Amano, S., et al., 1997) and in industrial systems such as IntelliCage of NewBehavior (New Behavior, 2013). However, these systems are limited in the type of information that they can supply and they cannot detect complex behaviors. For instance, light or vibration sensors are limited to general motion and tracking without the ability to identify the nature of the motion. Moreover, these systems are costly, require maintenance, and may be intrusive. Computer vision presents a promising alternative which is non-intrusive, relatively cheap and mostly universal in the sense that it could detect and classify all

---

3. FARAH, R., LANGLOIS, J.M.P. and BILODEAU, G.A. Computing a Rodent's Diary. Submitted to *IEEE Transactions on Image Processing (TIP).*

the behaviors presently requiring several sensor types.

Rodent monitoring presents great challenges. Rodents are kept in cages that are stacked on shelves which constrains camera localization. In a biomedical laboratory, the illumination is seldom uniform. Cages are host to all sorts of reflections due to their material, and their floors are covered with bedding to ensure animal comfort. The contrast between the animal colors and its surrounding color can be limited. Finally, the rodent's shape is extremely deformable, making modeling its body a difficult task.

In this paper, we propose a method for detecting and classifying three behaviors for rodents in a typical biomedical environment. The three behaviors are:

> Exploring: the rodent is moving around its environment or it is exhibiting actions such as sniffing or sensing with its whiskers.
>
> Rearing: the rodent is in an elevated position with its front paws above the ground.
>
> Static: the rodent is motionless. This may be due to the rat resting, sleeping or being motionless as part of a seizure behavior.

Those three behaviors were chosen among a list of medium-level behaviors that are of interest to our partners at Sainte-Justine Hospital to measure the effect of certain diseases or drugs on the physical or mental state of rodents. Other researchers have also focused on these behaviors. For instance, Rudenko et al. (2009) measured the number of rearing instances and the distance travelled by R6/2 mice suffering from Huntington's disease to observe their hypoactivity patterns. Gibbs et al. (2011), in a study of long term consequences of prolonged febrile seizures, identified arrest of movement or freezing as one of the manifestation of seizure in rats.

The proposed method uses the size of a rodent's Motion History Image (MHI), its centroid position and the output of a Multiple Classifier System (MCS) to detect and classify exploring, rearing and static. The MCS combines the outputs of two support vector machines (SVMs). The first SVM classifier uses the normalized height of the rodent's MHI as feature. The second SVM classifier uses the Histograms of Gradients (HOG) of the rodent's MHI. The contributions of this paper consist of:

This work is funded by CRSNG (Conseil de recherche en science naturelles et en génie du Canada) and the FQRNT (The fonds de recherche du Québec – Nature et technologies).

a) A new method to use MHI features to detect and classify animal behavior.

b) A new rule to combine the output of the two SVM classifiers. The rule is based on dynamic instantaneous weights.

This paper is organized as follows. Section 5.2 describes related work on animal behavior detection. Section 5.3 provides background on MHI and MCSs and Section 5.4 motivates our approach. Section 5.5 describes the proposed methodology, while Section 5.6 describes the classifier training methodology, the datasets and the experimental results. Section 5.7 concludes the paper.

5.2 Related work

Several projects in the litterature are specialized in animal behavior detection. This is justified by the large difference in anatomy and behavior between humans and animals. Rodents have few distinguishable features, their limbs are almost unnoticeable and some of them exibit brisk motion (Belongie, S., et al., 2005).

Nie et al. (2008) used a computer vision technique to detect and classify rat behaviour in forced swim tests. The method detects immobility, climbing and swimming. A combination of binarization, frame differencing, centroid calculation and filtering is used to monitor a dark mouse in a transparent container filled with water and positioned above an IR illuminator. In another work (Nie, Y., et al., 2011), the same authors, used a combination of frame binarization, frame differencing, centroid calculation, edge and contour extraction and a filtering method they developed in (Nie, Y., et al., 2009), to detect six behaviors in mice (Moving, rearing, immobility, head grooming, left-side scratching and right side scratching). These approaches require the use of a High Frame Rate camera.

Dollar et al. (2005), used 3D spatio-temporal features that rely on gradients in addition to a k-means classifier to detect drinking, eating, exploring, grooming and sleeping. Jhuang et al. (2010a) also used 3D spatio-temporal features. They combined them with ten other position and velocity based features as well as a Hidden Markov Support Vector Machine to detect eight behaviors: eating, drinking, grooming, hanging, micro-moving, rearing, resting, and walking. The method was tested on long video sequences to demonstrate its stability. However, it requires background subtraction as a prerequisite.

Market solutions have also been provided for rodent monitoring. We are aware of only one that provides a computer vision solution. CleverSys (Wilks, S.L., et al., 2004) identifies 21

different behaviors. However, for efficient results, the system requires the use of a special system that provides adapted lighting and a uniform white background (CleverSys, 2013). This is because the system uses background subtraction which requires constant uniform lighting and a certain level of stability in the background.

All of the above methods use binarization or a background subtraction to isolate the rodent before identifying its behavior. This induces constraints on the environmental settings and on some parameters of the ongoing biomedical experiment. For instance, due to the binarization, a high contrast is needed between the animal and its background for the method to be efficient. In fact, two of these methods (Dollar, P., et al., 2005) (Jhuang, H., et al., 2010a) are applied only on dark mice. In addition, with the exception of the works by Jhuang et al. and of Wilk et al., these methods were not proven for long video sequences (Jhuang, H., et al., 2010a).

5.3 Background and Motivation

5.3.1 Motion History Image

The MHI is robust in suppressing static objects in the scene and is able to preserve short duration complex movements (Yau, W., et al., 2006). These characteristics are crucial for rodent monitoring, because these animals tend to exhibit erratic and, in some cases, abrupt motion. This behavior is outlined when rodents are grooming, or when they are having seizures. In addition, MHI is invariant to the target color (Yau, W., et al., 2006) and to background color. This is important because rodents have different colors and are constantly subject to color distribution variation. In an environment where illumination is not uniform, or when the rat is crossing from one illumination zone to another, its color distribution changes. Color distribution change is also caused by the constant deformation of the rodent's body as it moves around.

MHI is a map that represents the presence of motion at different instants in time (Ahad, Md A. R., et al, 2012). Every pixel in the MHI represents motion imprint at that pixel. Bright and dark pixels represent recent and older motion, respectively. Bobick and Davis (1996) were the first to use MHI to recognize actions or behaviors. The authors used an image differencing technique to compute the motion pixels used to update the MHI. The image difference technique was chosen for its popularity (Davis & Bobick, 1997). They also used the Seven Hu moments (Hu, M., 1962) and the Mahalanobis distance (Therrien. C., 1989) to describe the MHI and classify the behavior. The authors later used background subtraction instead of im-

age differencing to decrease noise in the resulting MHI (Bobick & Davis, 2001). Ahad et al. (2010) argued that image differencing and background subtraction do not produce suitable MHIs. The image differencing method leaves holes in the MHI and background subtraction is not always effective (Ahad, Md A. R., et al, 2010). Instead, they used optical flow, the Seven Hu moments and the k-nearest neighbors algorithm to construct, describe and classify the MHI, respectively. The original MHI presents a self-occlusion problem (Ahad, Md A. R., et al, 2010). When an action presents two motions going in opposite directions, the later part will occlude the earlier one. Ahad et al. (2010) proposed a method with four MHIs to solve this problem. Each MHI represents a direction of the motion. The optical flow was used to calculate the MHIs. They also coupled the four MHIs with the Motion Energy Image (MEI). The MEI is calculated by thresholding the MHI over zero. The Seven Hu moments were calculated on the five images resulting in a 35-value vector feature to classify.

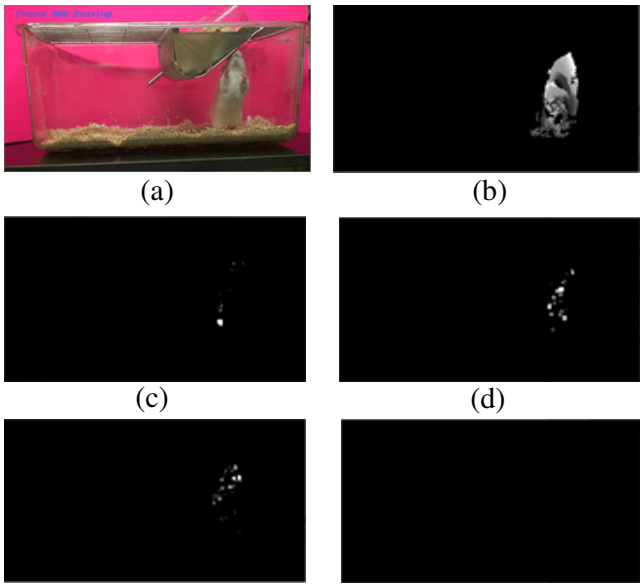


Figure 5.1 Two methods for computing the MHI of the frames leading to the frame in (a), (b) using frame differencing, (c-d) using optical flow [18]. (c) $\mathbf{MHI}_{\mathbf{x}}^{+}$, (d) $\mathbf{MHI}_{\mathbf{x}}^{-}$, (e) $\mathbf{MHI}_{\mathbf{y}}^{+}$, (f) $\mathbf{MHI}_{\mathbf{y}}^{-}$

When applying the optical flow based MHI on video sequences of rodents, the information provided is not sufficient. Figure 5- 1(b) shows an MHI computed using frame differencing on a video sequence of a rat. Figure 5-1 (c-f) shows the four MHIs computed using the optical flow as in Ahad's et al. (2012) article on the same frames. The information deficiency in Figure 5-1 (c-f) with respect to Figure 5-1(b) is clear. Figures 5- 1 (c-f) contain much less information than Figure 5- 1(b). The number of useful pixels is considerably reduced between Figure 5- 1(b) and the others. Figure 5- 1(f), which represents the negative motion in the y direc-

tion, contains no useful information. These results are explained by the fact that the rodent's body is extremely deformable. This makes its texture very variable and hard to model. Rodents also tend to have less texture with respect to their environment. This excludes optical flow features from being formed or maintained on the rodent's body (Farah, R., et al., 2013) and disadvantages any method that uses optical flow on rodents (Dollar, P., et al., 2005).

The exploring, rearing, and static behaviors of rodents are not prone to self-occlusion as addressed in (Ahad, Md A. R., et al, 2012) given that these actions are one-directional. However, self-occlusion can still be caused by the fact that the body of the rodent is not shaped in rigid parts that move together maintaining the same dimensions. This may cause the motion of one part of the rodent to occlude the motion of another. This problem is addressed in Section 5.4.2.

5.3.2 Multiple Classifier Systems

An MCS is a classifier system where the outputs of several base classifiers are combined to estimate the class of a given object. MCSs are often adopted for their ability to simplify the combination of several features. MCSs are also known for their performance which exceeds that of individual classifiers in the best case (Rahman & Fairhurst, 2003). In the worst case, their performance is similar to the best classifier (Menahem, E., et al., 2009). When several features are considered, these features are combined into one longer feature. The numerical values of each feature are normalized to fit in the range [0,1] (Kuncheva, L. I., et al., 2001) to give equal weight to each one. It is also possible to weigh each feature differently (Xing, H.J., 2012). Appropriate weight selection can be a tedious and complex process to achieve the right balance. In this work, we set the MCS to to take advantage of the strongest feature at each instance of its usage.

Several papers in the literature have studied and compared MCSs. Kunchva et al. (2001) separated MCS in two categories: classifier selection and classifier fusion. Classifier selection rules are based on the assumption that each of the base classifiers is expert in a local area of the feature. When a feature is presented whose value is in a certain local area, the output of the corresponding classifier is advantaged over the others. Classifier fusion rules are used with the assumption that all classifiers are expert in the whole feature space and some fusion rule is used to combine their outputs. Zhang and Duin (2011) further separated the fusion MCS into homogeneous and heterogeneous classifiers. Homogeneous classifiers consist of the same classifier that is trained with different sets of data, whereas, heterogeneous classifi-

ers consist of different classifiers that are trained with the same dataset. Zhang and Duin also categorized the fusion-based MCS into fixed and trainable combiners. Fixed MCS combiners consist of combiners such as maximum, mean, and majority voting, while trainable combiners consider the output of each of the base classifiers as a new meta-feature. Any classification algorithm could be used to estimate the final class (Kuncheva, L. I., et al., 2001) (Zhang & Duin, 2011). Finally, the rule choice should be based on the nature of the data and the classifiers involved. After analysing and discussing ten different combination rules, Duin and Tax. (2000) concluded that there is no universally efficient rule.

In this paper, we propose a simple fusion rule that combines the output of SVM classifiers. This particular fusion rule is used for its simplicity and ease of application in this case. It was also chosen for its ability to allow instant dynamic weights that favor the strongest output at a given instance under the assumption that the strongest output is the most accurate.

5.4 Proposed Methodology

We now describe the method used to detect and classify exploring, rearing and static in rodents.

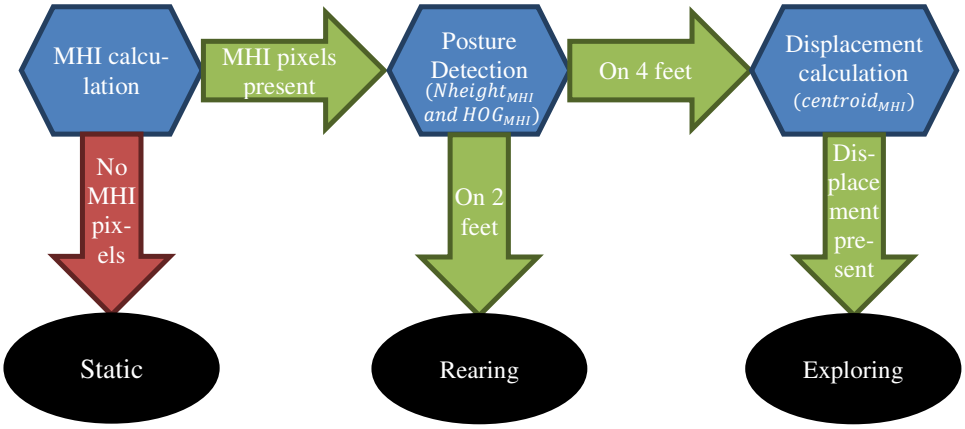


Figure 5.2 The strategy used to detect and distinguish the three stages

5.4.1 Strategy:

The proposed strategy is illustrated in Figure 5- 2. First, the MHI of the sequence is computed. When there are no MHI pixels, then the rodent is assumed to be in the static state. Otherwise, an MCS is used to determine the posture of the rodent. If the classifier returns an “on two feet” verdict, then the rodent is assumed to be in the rearing state. If the rodent is classi-

fied as “on four feet” and a displacement exists, then the rodent is assumed to be in the exploring state. A displacement is calculated as a displacement of the centroid of the MHI by a distance that is 10% of its size in the x- or the y-dimension.

5.4.2 Motion History Image:

The MHI is computed for each frame as described in (Bobick & Davis, 1996) with modifications to avoid self-motion occlusion. The MHI is computed in five steps:

a) A decay factor is imposed on the MHI $\mathrm{H}_\tau(\mathrm{t}-1)$ computed at $\mathrm{t}-1$ as shown in equation 1.

$$H_\tau(x,y,t) = max\{H_\tau(x,y,t-1) - \delta,\ 0\}, \tag{5.1}$$

where $\mathrm{H}_\tau(\mathrm{x},\mathrm{y},\mathrm{t})$ is the MHI value at time t at coordinates x and y, and δ is the decay parameter.

The motion presence map $\mathrm{D}'(\mathrm{x},\mathrm{y},\mathrm{t})$ at time t is calculated as in (Bobick & Davis, 1996) by subtracting two consecutive frames.

b) Then the MHI is updated according to the motion presence map $D'(x,y,t)$. In the original method (Bobick & Davis, 1996), the MHI is updated by setting the value at the coordinates x and y to τ if the corresponding value in $D'(x,y,t)$ is greater than zero as in equation 5.2.

$$H_\tau(x,y,t) = \begin{cases} \tau & if\ D'(x,y,t) > 0 \\ H_\tau(x,y,t-1) & otherwise \end{cases}, \tag{5.2}$$

where τ is the duration of the action. The duration of the action corresponds to the number of frames used to construct the MHI.

However, this method is prone to MHI self-occlusion as stated in (Ahad, Md A. R., et al, 2012). To avoid self-occlusion, we modify equation 2 so that the MHI at time t and coordinates x, y is updated only if its value is equal to zero. This is shown in equation 3.

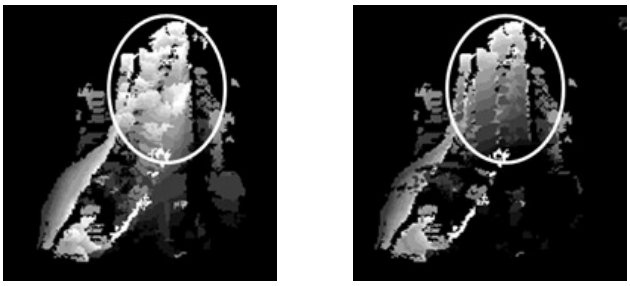
$$H_\tau(x,y,t) = \begin{cases} \tau \times D'(x,y,t) & if\ H_\tau(x,y,t-1) = 0 \\ H_\tau(x,y,t-1) & otherwise \end{cases}, \tag{5.3}$$

Figure 5. 3 shows the difference between the two methods. Figure 5- 3 (a) shows the MHI using Bobick and Davis’ original method and Figure 5- 3 (b) shows the method described in

this paper. The ellipses indicate the region where self-occlusion reduction is most apparent. In Fig 3 (a), the bright colors are dominant and they cover regions where in Fig 3 (b) the history of the movement is more apparent by the gradual change in color from dark to bright.

c) The frame is then subjected to morphological operations to eliminate noise blobs. These blobs are generated by reflections on the cage and motion from the mulch.

d) The largest blob in the MHI frame is chosen to represent the rodent. We assume that the rodent is the only dynamic entity in the scene and most likely to be the one to generate the largest blob.



(a) (b)

Figure 5-3 Difference between (a) the MHI method proposed by Bobick and Davis et al. (1996) and (b) the method described in this paper.

5.4.3 Features

Two features were considered to detect the three targeted behaviors: the normalized height of the MHI and the Histogram of Gradients of the MHI.

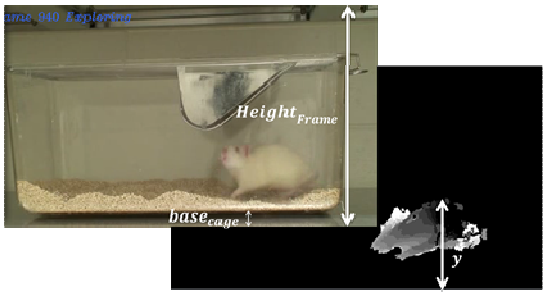


Figure 5.4 Parameters used to calculate the normalized MHI height

In this paper, the MHI's normalized height ($Nheight_{MHI}$) is defined as the y-coordinate of the highest point in the MHI. The MHI's height is normalized to account for the dimension variation in each rodent. $Nheight_{MHI}$, is calculated as follows:

$$Nheight_{MHI} = \frac{Height_{Frame} - y - base_{cage}}{width_{animal}}, \quad (5.4)$$

Where $Height_{Frame}$ is the height of the frame, y is the y-coordinate of the highest MHI pixel that belongs to the rodent and the $base_{cage}$ is the lowest pixel that belongs to the floor of the cage in the frame. $width_{animal}$ is the largest width recorded for the animal in the given video. This measure is usually recorded when the rodent is walking. This is when the rodent is most stretched out vertically. The maximum height was not considered because large rodents may not be able to stretch to full heights when the height of the cage is inferior to theirs. $width_{animal}$ could be substituted by the actual animal width or height if measurements were taken before the experiment. Figure 5- 4 shows the different measurement used.

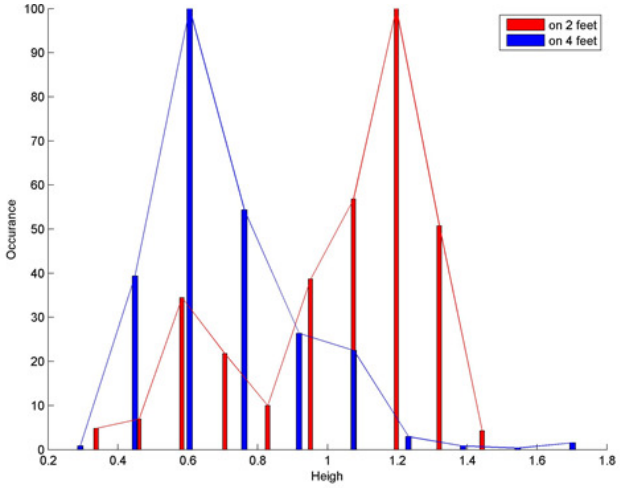
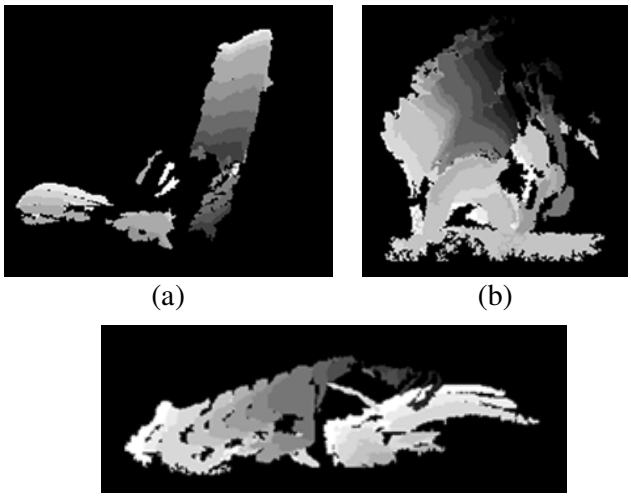


Figure 5.5 Height distribution

$Nheight_{MHI}$ is considered as a feature because height is a very strong indicator of the posture of the rodent. In fact, when the MHI is high the rodent is probably on two feet and consequently in a rearing state. Figure 5- 5 shows the height distribution of $\text{Nheight}_{\text{MHI}}$ calculated with respect to two postures (on two feet, on four feet) using a training set. $Nheight_{MHI}$ can be used to achieve some level of discrimination between the two postures, but it is not sufficient given that the two distributions are not completely separated.

Histograms of gradients (*HOG*) (Dallal & Triggs, 2005) are considered due to their ability to model texture. The MHI texture is a strong indicator of certain behaviors. Figure 5- 6 (a, b and c) represents a rat that rises on two feet, going down on four feet and walking. We can distinguish the various strata patterns that every action imprints on an MHI. For instance, in

Figure 5- 6 (a), we can distinguish, on the right side of the frame, the ascending pattern that marks the different stages that it took the body of the rat to rise. Notice also the color change from dark at the bottom, to mark oldest motion, to bright at the top to mark newest motion. On the left side of the frame, the motion of the rat's tail produces the same pattern.



(a) (b)

(c)

Figure 5.6 MHI representing different types of motion, a) a rat getting on two feet, b) a rat going down on 4 feet, c) rat walking

The HOG are computed as described in (Farah, R., et al., 2013) on the rodent MHI blob. The HOG algorithm described in (Dallal & Triggs, 2005) considers ROIs with variable dimensions where the grids have fixed dimensions and the cells have dynamic dimensions. This is useful because the MHI is continuously changing dimensions. The dimension of the grid is 4×4. Nine bins are used as suggested by the author of the original article (Dallal & Triggs, 2005) which results in an 81-value feature.

5.4.4 Multiple Classifier System

As described in Section 5.3.2 MCS present an attractive scheme to combine several features. For this purpose we propose a fusion rule for the MCS. The fusion rule is shown in equation 5.5.

$$resp_{MCS}^{t} = w_1^t \times dec_1^t + w_2^t \times dec_2^t + \cdots + w_n^t \times dec_n^t, \qquad (5.5)$$

The weights w_n^t at each instant t are calculated in such a way as to strengthen the contributions of the strong base decisions (dec_n^t) to the response of the MCS at t. In other words, for each decision instance t, the weight attributed to each base classifier is proportional to the

certainty of the decision of that base classifier at that instance t. This rule is simple and allows a higher weight for the assumed most accurate base classifier at each instance.

Our MCS uses two base classifiers: the first classifier (SVM_{height}) is an SVM that classifies $Nheight_{MHI}$.The second classifier (SVM_{HOG}) is an SVM that classifies the HOG of the MHI.

To calculate the posture at instance t based on the responses ($resp^t_{SVM_{height}}$,$resp^t_{SVM_{HOG}}$) of the SVM classifiers, two thresholds ($vt_{SVM_{heigh}}$, $vt_{SVM_{HOG}}$) are used. We define a base classifier estimation (dec^t_n) as the difference:

$$dec^t_n = resp^t_n - vt_n, \tag{5.6}$$

At each instance t, to interpret the response of the n^{th} two-class classifier, we consider the value of dec^t_n. Given that each class is assigned a scalar value, if dec^t_n if negative, then the posture belongs to the class with the low value. Otherwise, if dec^t_n is positive then the posture belongs to the high value class. In addition, we assume that the further the response $resp^t_n$ of a base classifier is from its corresponding threshold vt_n, the more assertive its decision dec^t_n is. Consequently, we calculated the instant weight attributed to each base classifier as the absolute value of its decision.

In this case, the fusion rule to determine the posture at each instance (frame) becomes:

$$\mathrm{resp}^t_{MCS} = w_{height} \times dec^t_{SVM_{height}} + w_{HOG} \times dec^t_{SVM_{HOG}}, \tag{5.7}$$

where

$$\mathrm{w_{height}} = \mathrm{abs}\left(\mathrm{resp^t_{SVM_{height}}} - \mathrm{vt_{SVM_{height}}}\right), \tag{5.8}$$

$$dec^t_{SVM_{height}} = resp^t_{SVM_{height}} - \mathrm{vt}_{SVM_{height}}, \tag{5.9}$$

$$w_{HOG} = abs\left(resp^t_{SVM_{HOG}} - \mathrm{vt}_{SVM_{HOG}}\right), \tag{5.10}$$

and

$$dec^t_{SVM_{HOG}} = resp^t_{SVM_{HOG}} - \mathrm{vt}_{SVM_{HOG}}, \tag{5.11}$$

The posture is evaluated as

$$posture = \begin{cases} on\ two\ feet & \text{resp}_{\text{classifier}} < 0 \\ on\ four\ feet & otherwise \end{cases}, \quad (5.12)$$

where $posture$ is the final decision of the MCS.

We used the OpenCV libraries for the implementation of the SVM classifiers (Willow Garage, 2013).

5.5 Classifier training, data Sets and results

In this section we detail the parameters of the MCS used. We also describe the datasets used for training the system and the sequences used to for testing and evaluation. Finally, we present experimental results.

5.5.1 Training Datasets for the classifier

Sixty-one training dataset samples were extracted from three different video sequences of a single rat in a cage. The three video sequences where chosen to represent different background colors. The rats in the video sequences are all white. The background colors used are pink and white (the natural colors of the wall behind the cage).

The threshold (vt_n) associated with each base classifier is calculated as the threshold that minimizes the number of false decisions for that particular classifier using the training dataset. For this purpose, the number of false decisions is calculated for a range of threshold values and the best one is retained. The resulting thresholds are $\text{vt}_{SVM_{height}}$= 3.0 and $\text{vt}_{SVM_{HOG}}$= 3.5.

5.5.2 Test Sequences for the algorithm

We tested the proposed methodology on four different video sequences of rats and mice, listed in Table 5.1. These four sequences are different from the sequences used for training. The first three sequences are sequences of different white rats in cages. These sequences were recorded by us in a medical laboratory where experiments aiming to study seizures in rats are conducted. The cages were transparent, stored on shelves and covered with beddings. The illumination was provided by fluorescent lamps fixed to the ceiling. The camera was mounted in such a manner as not to disturb the ongoing experiments. One cage with one rat was recorded at a time. The last sequence was extracted from the dataset provided in (Jhuang, H., et al., 2010b). It is a sequence of a black mouse in a cage. The cage is transparent and its floor is

covered with bedding.

Table 5.1 Test sequences

Sequence	Rodent	Rodent Color	Background Color	Frame Number
Sequence 1	Rat	White	Pink	6770
Sequence 2	Rat	White	Pink	17881
Sequence 3	Rat	White	No Background	7072
Sequence 4	Mouse	Black	No Background	68227

5.5.3 Experimental settings and results

During the experiments, the decay factor δ, (equation 1), was set to 1 as used in most of the state of the art. The action duration parameter τ, (equation 3), was set to 13 because it was noted (according to our training dataset) that the smallest time a rat stays on 2 feet is 13 frames. This minimum was adopted because it was enough to account for a rearing behavior. It also ensures that two consecutive actions would not occlude each other.

A linear kernel was used for SVM_{height}, the SVM used to estimate the posture of the rodent using the height of the MHI. A radial basis function kernel was used for SVM_{HOG},the SVM used to estimate the posture of the rodent using the HOG of the MHI. These choices are based on the experimental results shown in Table 5.2. The linear and the Radial Basis Function (RBF) kernels were applied to SVM_{height} and SVM_{height}. The testing data set was used to determine which combination is best. Table 5.2 shows that SVM_{height} performed better with the linear kernel, while SVM_{HOG} performed better with the RBF kernel.

Table 5.2 Kernel test results, for 30 tests

	Number of false classifications	
	Linear kernel	**RBF kernel**
$\mathrm{SVM_{height}}$	2	3
$\mathrm{SVM_{HOG}}$	7	6

To evaluate the results objectively, we calculated the confusion matrix for each sequence and the average of the resulting four confusion matrices. The sequences were evaluated with respect to a ground truth built by us. The results for the five sequences are summarized in Figure 5- 7.

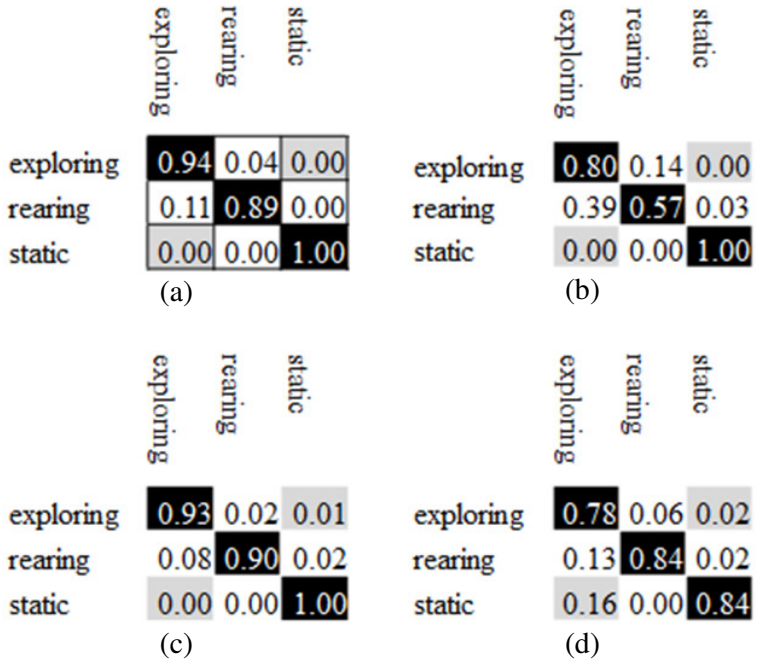


Figure 5.7 Confusion Matrices for a) Sequence 1, b) Sequence 2, c) Sequence 3, d) Sequence 4

The results in Figure 5- 7 are consistent with what is acceptable in animal monitoring in bio-medical laboratories (Nie, Y., et al., 2011). Rearing was less accurately estimated due to the fact that when a rodent is in a rearing position, it can stay motionless for some time. However, the rodent would still exhibit some motion in its lower body due to the motion of the tail or some fidgeting around its posterior feet. This results in an MHI that covers only the lower part of the body of the rodent. In this case the MHI has a small height and is classified as on 4 feet.

Several snapshots of the labeled sequences are shown in Figure 5- 8.

Even though the Static behavior was perfectly identified for the first three sequences, the decrease in sequence 4 is due to the low resolution of the video. The low resolution prevents the morphological operations to reduce the noise without suppressing the MHI of the mouse. Occasionally, the largest MHI blob was associated with noise and not the mouse itself, thus when the rodent was actually static some erroneous activity was detected.

In order to locate our results with respect to the state of the art, we considered results reported in other projects. An exact comparison cannot be done due to the fact that the videos and their ground truths are different. In addition, the behaviors considered in those articles are not defined in the same manner as the ones defined in this article. We considered the works by Dollar et al. (2005) and Jhuang et al. (2010a). We also mention results reported for CleverSys by

(Jhuang, H., et al., 2010a). We assumed that Walking, Resting and Rearing in (Jhuang, H., et al., 2010a) and CleverSys correspond to Exploring, Static and Rearing, respectively, as defined in this work. Jhuang et al. (2010a) tested their algorithm and the CleverSys product on 12 different videos of black mice in transparent cages. In addition, we assumed that Exploring

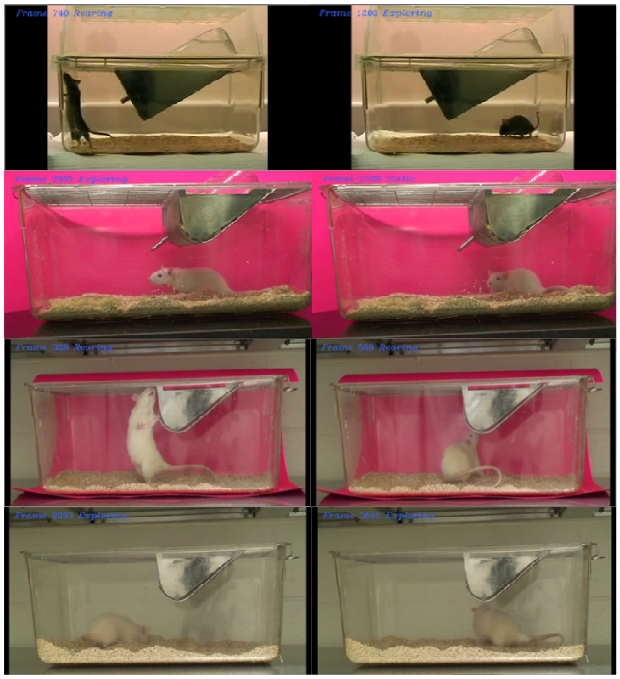


Figure 5.8 Snapshots from the test sequences showing different behaviors.

and Sleeping as defined by (Dollar, P., et al., 2005) correspond to Exploring and Static, respectively, as defined in this work. Dollar et al. (2005) tested their algorithm on seven film clips.

Table 5.3 The ratio of correct identifications

	Exploring	**Rearing**	**Static**
CleverSys	0.76	0.33	0.87
(Jhuang, H., et al., 2010a)	0.68	0.63	0.83
(Dollar, P., et al., 2005)	0.86	N/A	0.87
This work	0.84	0.73	0.96

Table 5.3 suggests that our results compare favorably with results reported in the state of the art and with industrial products even though no environmental constraints were applied. For example, the white rats were monitored while the background was also white, decreasing the contrast between the rodent and the background. Furthermore, the illumination was not controlled or uniformly distributed in the region of interest.

The training set comprised only of white rats while it was used on detecting and classifying the three behaviors for white rats and black mice alike. This independence with respect to color is due to the MHI which is color insensitive. Furthermore, the system was tested on video sequences of short and long durations. It performed equally well in both cases. This demonstrates that the system maintains its stability when applied to long video sequences.

Furthermore, the system was equally performing on sequences of long or short duration. This demonstrates the stability of the system, given that the results did not deteriorate when tested on long video sequences.

5.6 Conclusion

In this article, we proposed a method that detects and classifies three states in rodent behaviors: exploring, rearing and static. The method use MHI-based features with a Multiple Classifier System where at each instance the base classifier with the strongest decision is advantaged. The method was tested under typical biomedical lab conditions, in such a way as not to disturb the ongoing experiments. The proposed method achieves an accuracy of 87%.

CHAPTER 6 A COMPUTATIONALLY EFFICIENT IMPORTANCE SAMPLING TRACKING ALGORITHM[4]

Abstract — This paper proposes a computationally efficient importance sampling algorithm applicable to computer vision tracking. The algorithm is based on the CONDENSATION algorithm, but it avoids expensive operations that are costly in real-time embedded systems. It also includes a method that reduces the number of particles during execution and a new resampling scheme. Our experiments demonstrate that the proposed algorithm is as accurate as the CONDENSATION algorithm. Depending on the processed sequence, the acceleration with respect to CONDENSATION can reach 7× for 50 particles, 12× for 100 particles and 58X for 200 particles.

6.1 Introduction

Since their introduction in computer vision in 1996 (Isard & Blake, 1996; Isard & Blake, 1998a), particle filters have grown in popularity and have been applied to several specific applications. Among these applications, we can cite animal tracking (Isard & Blake, 1998b; Black & Jepson, 1998; Chellappa & Zhou, 2003; Yogesh, R., et al., 2005), human gesture recognition (Isard & Blake, 1998b; Black & Jepson, 1998; Bretzner, L., et al., 2002), human face recognition (Chellappa & Zhou, 2003), and tracking sport players on court (Okuma, K., et al., 2004; Vermaak, J., et al., 2003). The applications that use particle filters are not limited to the visible spectrum, but extend to infrared (Bilodeau, G. A., et al., 2012), laser (Meier & Ade, 1999; Fox, W. B., et al., 1999; Kwok, C., et al., 2004), and sonar (Fox, W. B., et al., 1999; Kwok, C., et al., 2004; Dellaert, F., et al., 1999) measurements in robotic vision. Particle filters have gained such popularity because of their simplicity and generality (Isard & Blake, 1998c). Furthermore, particle filters have proven to be robust trackers in the case of partial occlusion (Isard & Blake, 1996; Isard & Blake, 1998a).

However, particle filters are not ideal trackers. They present a compromise between tracking precision and computational load. Particle filters use particles to estimate the location probability of the target. In order to have a precise tracker, the number of particles should be high

---

4. FARAH, R., LANGLOIS, J.M.P. and BILODEAU, G.A., FENG, Q. SAVARIA, Y. A Computationally Efficient Importance Sampling Tracking Algorithm. Submitted to *Machine Vision and Applications (MVAP)*.

so that a large number of possible locations are covered. In particular, the CONDENSATION algorithm requires many complex operations that present several computational bottlenecks. These include Gaussian random value generation and floating point manipulations. In an embedded implementation, this could cause high computational complexity and energy dissipation. In this paper, we target these problems and propose solutions in order to make the implementation of the CONDENSATION algorithm computationally efficient. It can then be more suitable for hardware implementation when resources are limited, for instance on smart cameras. Our efforts are concentrated at the algorithmic level.

Our contributions revolve around proposing a tracking algorithm that provides competitive processing quality while requiring reduced computational complexity as compared to the popular previously reported method (Isard & Blake, 1998a). We propose means to dynamically reduce the average number of particles effectively processed at each time step. These simplifications have a dramatic impact on software acceleration, and could also facilitate hardware implementations. We show that our algorithm maintains roughly the same accuracy. A detailed complexity analysis is provided, showing that the proposed means to interrupt sampling reduces the average number of processed particles by 85% of that required in the CONDENSATION algorithm. The first contribution is a sampling scheme that reduces time complexity while preserving adequate precision. Computation time reduction leads to lower energy consumption, which is an important aspect for autonomous systems. The second contribution concerns resampling and consists of a simplified method to assign the number of particles generated at time $t + 1$, from the set of particles at time t.

This paper expands on our previous work (Farah, R., 2011). In this paper, we propose a new method to set some of the algorithm's parameters. We apply the algorithm to a set of videos that cover a wider range of difficulties and contexts. Finally, we perform a more extensive analysis of the results.

The paper is structured as follows. Section 6.2 reviews previous work. Section 6.3 consists of a description of the proposed algorithm which will be called SETA (Simplified Efficient Tracking Algorithm) through the rest of the paper. Section 6.4 describes the implementation and test procedures. Section 6.5 gives results and presents an analysis of the methodology, and section 6.6 concludes the paper.

6.2 Previous Work

Particle filters were first introduced in computer vision with the CONDENSATION algorithm by Isard and Blake (1996) (1998a). In the CONDENSATION algorithm, the state of the target object at time t is denoted $\boldsymbol{x}_t$. $\boldsymbol{x}_t$ is estimated by N particles at coordinates $\rho_t^{(i)}$and each of those particles is characterized by a weight $w_t^{(i)}$. In order to estimate the state of the target at time $t+1$, new samples $\boldsymbol{\pi_t}$ at coordinates $\rho_{t+1}^{(i)}$ are generated from $\boldsymbol{x}_t$ using a random process that is biased by the weights of the particles at time t. Consequently, the sample at time t that has the best weight has a highest probability to be propagated, while the sample with the worst weight has a lowest probability to be propagated. Samples at time t are propagated to $t+1$ by applying a transformation that consists of two steps. One step is ruled by prior information and the other step consists of the addition of a normally distributed random value. After propagating the samples, they are weighted with respect to an observation $\boldsymbol{z}_{t+1}$ made at time $t+1$. The pseudo code for the CONDENSATION algorithm is shown in Figure 6.1.

Several other work related to object tracking are based on the CONDENSATION algorithm. One class of work deals with the problem that results from processing a large number of particles and proposes several solutions to solve it. To decrease the number of particles, two strategies are introduced. The first strategy consists in proposing a new sampling scheme in order to have better coverage of the search space with a smaller number of particles. In (Isard & Blake, 1998b) and (Black & Jepson, 1998), importance sampling is proposed to replace factored sampling, which is used by CONDENSATION, because importance sampling has less tendency to cluster samples. This algorithm was called ICONDENSATION. Philomin et al. (2000) proposed to substitute a conventional pseudo random generator with a random generator based on a Sobol sequence, which generates a better random distribution. Furthermore, Maggio & Cavallaro (2005) used a mean shift tracker on all the particles after resampling in order to improve the particle quality. Deutscher et al. (2000) proposed a particle filter that uses a simulated annealing step in order to propagate samples with the intention of producing particles that have a higher probability of representing the target. Fox et al. (2003) proposed a particle filter for robot localisation. The particle filter uses a dynamic number of particles. The number of particles varies whether the state uncertainty is high or low. It is also calculated as the number of particles needed to achieve, with high probability, a good approximation of an arbitrary discrete probability distribution at the concerned time step.

The second strategy consists in proposing new propagation models that give a better estima-

tion of target dynamics. A better estimation of the target dynamics can reduce the search space and consequently can reduce the number of particles. For instance, in (Isard & Black, 1998), several propagation models are used in order to better represent the dynamics of a writing hand. Okuma et al. (2004) and Wang et al. (2005) proposed a "boosted particle filter" which consists of a particle filter that uses the Ada boost algorithm at the particle propagation stage. Yong & Yunqiang (2001) and Li et al. (2003) used an Unscented Particle Filter for target tracking. The Unscented Particle Filter consists of a particle filter that uses an Unscented Kalman Filter to propagate the particles. Li et al. (2003) also described a Kalman Particle Filter that uses a Kalman Filter in the propagation step. Saboune and Laganiere (2009) used a mixture of several sampling methods to generate the particles.

// Resampling (or selection)

1. Given $\{x_t^{(0)}, x_t^{(1)}, \dots, x_t^{(i)}, \dots x_t^{(N-1)}\}$, select N samples as follows:

For j = 1:N

2. Generate a uniformly distributed random number $r_j \in [0,1]$

3. Select the sample $\pi_t^{(i)}$ from $\{x_t^{(0)}, x_t^{(1)}, \dots, x_t^{(i)}, \dots x_t^{(N-1)}\}$ where $i = \{1, \dots N-1 / \ c_t^i \geq r_j\}$. c_t^i is the *cumulative probability of the* $\pi_t^{(i)}$

4. Set $x_{t+1}^{(i)} = \pi_t^{(i)}$

End for

// Prediction

5. Propagate the N particles $\{x_t^{(0)}, x_t^{(1)}, \dots, x_t^{(i)}, \dots x_t^{(N-1)}\}$

// Measurement

6. for the N particles $x_{t+1}^{(0)}, x_{t+1}^{(1)}, \dots, x_{t+1}^{(i)}, \dots x_{t+1}^{(N-1)}$

7. *Calculate the weight of* $x_{t+1}^{(i)}$ $(W_{t+1}^{(i)})$

8. *Normalize the weights so that* $\sum_{i=0}^{N-1} W_{t+1}^{(i)} = 1$

9. *And calculate the cumulative probability:*

$$c_{t+1}^0 = 0,$$
$$\vdots$$
$$c_{t+1}^n = c_{t+1}^{n-1} + W_{t+1}^{(n)},$$
$$\vdots$$

//Estimation

10. $\varepsilon[x_{t+1}] = \sum_{n=1}^{N} W_{t+1}^{(n)} \pi_{t+1}^{(n)}$

Figure 6.1 The CONDENSATION algorithm's pseudo code

In order to reduce the number of particles needed for adequate tracking, most of these work introduced new steps to the algorithm with more complex computations. As argued in (Li, P., et al., 2003), reducing the number of particles does not always lead to faster trackers, because

the modifications done to the original filter tend to introduce additional computations (as in the Unscented Kalman Particle Filter).

From the embedded system implementation perspective, Velmurugan et al. (2007) and Medeiros et al. (2008) (2010) presented solutions for implementing the CONDENSATION algorithm on different types of technologies for real-time video tracking applications. The methods suggested in these papers used different observation models, different measurement models, or different propagation models to optimize their tracking results. However, what these methods have in common is that they all use the same resampling method used in (Isard & Blake, 1998a) that requires computations done on all N filter particles. N is usually a large number to ensure good tracking. For instance, Isard and Blake (1998a) used $N = 500$ in one of their experiments. Li et al. (2008) tried to reduce the value of N while maintaining good quality tracking by introducing a mean-shift step into the particle filter. The mean-shift step clusters the particles closer to the higher probability zone. However, adding the mean-shift step introduces additional computational load.

6.3 Proposed Tracking Algorithm

SETA is a tracking algorithm based on the CONDENSATION algorithm for object tracking in videos. We propose a number of transformations to the algorithm that reduce computational complexity and that could simplify an eventual hardware implementation. We also propose a means to dynamically reduce the number of particles effectively processed, at each time iteration, while keeping a same number of maximum particles as the reference algorithm. These simplifications are implemented in software. It is shown in the rest of the paper that our algorithm maintains an externally apparent functionality that is very similar to the CONDENSATION algorithm while maintaining essentially the same tracking accuracy. The pseudo code for the proposed tracking algorithm is given in Figure 6.2 and details are given below.

6.3.1 Resampling

The CONDENSATION algorithm (Isard & Blake, 1998a) uses the Inverse transform which is biased by the weight of the particles (Figure 6.1– lines 1, 2, 3, 8 and 9). In its original implementation (Isard & Blake, 1998a), particles with better weights have a higher chance of being propagated than particles with worst weights. However, this operation requires a division for each particle to normalize the weights (Figure 6.1– line 8), in addition to a random value per sample to extract the sample and propagate it into the next generation (Figure 6.1– line 2).

The divisions and random value generations are very costly operations when considering real-time embedded system applications. Furthermore, both operations involve floating point values. To avoid random value generation and calculations using floating point values, and to minimize the number of divisions, the proposed tracking algorithm selects the best weighted p samples out of the N particles to form the seed of the next particle generation (Figure 6.2-line 1). The value of p is chosen in a way that satisfies a compromise between computational cost reduction and the tracking quality. This operation does not require any division or random value generation and generates no floating point values.

```
// Resampling (or selection)
1. Given {x_t^(0), x_t^(1), ..., x_t^(n), ... x_t^(N-1)}, select the p samples with the best weights
   {π_t^(0), π_t^(1), ..., π_t^(i), ... π_t^(p-1)}

//Prediction & Measurements
j= 1
while j <N & N_γ ≥ β
   2. π_t^(i) = get a sample from {π_t^(0), π_t^(1), ..., π_t^(i), ... π_t^(p-1)}
      as described in Section A
   3. x_(t+1)^(n) = π_t^(i)
   4. Propagate x_(t+1)^(n)
   5. Calculate the weight W_(t+1)^(n) of x_(t+1)^(n)
   6. if (W_(t+1)^(n) ≥ γ) increment a counter N_γ
      j = j + 1
end while

//Estimation
7. ε[x_(t+1)] = π_(t+1)^(n) | (W_(t+1)^(n) = min{W_(t+1)^(0), W_(t+1)^(1), ..., W_(t+1)^(n)})
```

Figure 6.2 The proposed algorithm pseudo code

To further reduce the number of divisions and random values generated, the number of particles calculated from each seed particle is pre-set according to their respective weights, as follows.

Given a list of p seed particles $\{\pi_0, \pi_1, \dots, \pi_{p-1}\}$ that are sorted according to their weights (best weight first, worst weight last), the number of samples generated from each seed $\{r_0, r_1, \dots, r_{p-1}\}$ is set according to (6.1) and (6.2).

$$r_0 + r_1 + \dots + r_{p-1} = N\,, \tag{6.1}$$

And

$$r_1 = r_0 - \alpha, \qquad r_2 = r_0 - 2 \times \alpha, \ldots,$$

$$r_{p-1} = r_0 - (p-1)\alpha, \tag{6.2}$$

$$r_0 \geq (p-1)\alpha, \tag{6.3}$$

where N is the total number of particles and α is a predefined value. Without loss of generality, assuming (6.3) is respected, we can always compute an integer value of N from any combination of integer values of α, p and r_0. Of course, the user should make sure the resulting N is acceptable (larger than, smaller than or close to a desired N_T). In that case, all the relevant parameters are integers, which simplifies computation and reduces storage space.

6.3.2 Prediction and Measurement

Particle processing is the most costly step in the CONDENSATION algorithm (Isard & Blake, 1998a). The weight calculation results in a heavy computational load and large memory space. In addition, the number of particles is fixed and is usually large to ensure good tracking (Figure 6.1– lines 5, 6, and 7). In contrast, SETA merges the prediction and measurement steps to allow reducing the number of particles when possible, while still ensuring good tracking quality and the possibility of saving computations (Figure 6.2- lines 5-7).

In order to reduce the number of generated particles, the particle weights at t+1 are calculated after each particle is built. As each new particle is generated, the method calculates the number of times a new particle has a weight better than a threshold γ at t+1. If this number es β, then particle generation is interrupted.

For some iterations, β will not be reached, which represents the worst case scenario from a computational complexity standpoint. This occurs when the target changes appearance or when the target is subjected to an occlusion. The particles resemblance to the template will decrease and their weights will degrade. In this case, new particles will continue to be generated until the maximum N is reached and the number of particles calculated by our method will be equal to the number of particles used by the CONDENSATION algorithm (Isard & Blake, 1998a). However, the earlier β is reached, the fewer the number of generated particles and the greater the computational load reduction. Note that when assigning the samples at t+1, a particle at t may be chosen several times or it may never be chosen.

6.4 Implementation and test procedures

In this work, we apply the SETA and CONDENSATION (Isard & Blake, 1998a) algorithms to video tracking. To enable fair comparisons, the same test conditions and features were used for both algorithms. In particular, the same observation model, weight calculation, and propagation model were used for both algorithms. These features and test conditions were chosen to facilitate implementation and to obtain moderate complexity, while maintaining good processing quality. Accordingly, and to distinguish between the implementation proposed in (Isard & Blake, 1998a) and the implementation proposed in this paper of the CONDENSATION algorithm, our implementation will be referred to as CONDENSATIONh. In our implementation, we adopted a convention that differs from the one used in the original CONDENSATION algorithm. In this implementation, lower weight particles are better and they are given higher priority. This is true for SETA as well as for CONDENSATIONh.

The generic SETA algorithm proposed in section 6.3 was adapted to video tracking and the significant implementation details of how this was done are as follows:

As in competitive video trackers to which we will compare (Deutscher, J., et al., 2000), (Li, P., et al., 2003), SETA and CONDENSATIONh algorithms process images in gray scale. In addition, gray scale reduces processing complexity.

The template model consists of a bounding rectangular region centered on the target in the first frame of each sequence. The region was selected manually. An automatic object detection method could in principle be used to select such targets. However, the aim of this section is to validate the tracking process, so no automatic process is applied.

It is of interest that the target could change in appearance and dimensions through the sequence. The variance in dimensions is very hard to predict a priori. In addition, when considering a hardware implementation in the future, memory space is allocated for the template model and for each sample. Fixed dimensions are preferred in the case of hardware implementations to allow more efficient memory management.

The template model is occasionally updated when the weight of the best particle falls below a certain threshold δ (which is considered better). The template model is updated to account for the changes in the grayscale distribution of the target or its target-to-rectangle surface ratio. Consequently, the update will promote good particles with smaller weights. In other words, when the target changes appearance, the template model will be updated to account for those changes. This will reduce the distance between the particles that are affected by this change of

appearance and the template model. Therefore, the chances to get particles that have weights better than γ are increased and accordingly the number of particles to be processed is more likely to be reduced. The restriction, set by the threshold δ, is used to ensure good quality tracking by avoiding updates when the tracking is not optimum.

Given $\mathcal{T}_t$, the template model, and $\mathcal{P}_t$, the particle with the lowest weight at time t, the model is updated as follows

$$\mathcal{T}_{t+1} = \zeta\mathcal{T}_t + (1-\zeta)\mathcal{P}_t, \tag{6.4}$$

where ζ is a user specified parameter.

The observation model consists of the intensity histogram. A sample consists of a rectangular region in the grayscale frame space of dimensions set before processing each benchmark according to condition 2. Intensity histograms are chosen for their robustness to rotation and scaling (Nummiaro, K., et al., 2003).

The weight of a particle, $W_p$is the city block distance between the intensity histogram, $\mathcal{H}_p$, of the particle and the intensity histogram of the template model, $\mathcal{H}_M$. The city block distance is chosen because of its simplicity and because the only operations involved are additions, subtractions, and an absolute value. It is calculated as

$$W_p = \sum_{j=0}^{K-1}\left|h_p[j] - h_M[j]\right|, \tag{6.5}$$

where K is the number of histogram bins.

Given that the weight of a particle represents the distance between the histogram of that sample and the one of the template model, the best weight is the smallest weight.

The propagation formula used in (Torabi, A., 2008) is applied. Again this formula is chosen for its simplicity and low computational requirements. The formula is:

$$\mathcal{X}_{t+1}^{(n)} = \mathcal{X}_t^{(n)} + \mu_{t+1}, \tag{6.6}$$

where $\mathcal{X}_t^{(n)}$ represents the coordinates of the nth particle at time t and μ_{t+1} is a Gaussian random variable that produces a translation in both the x and y directions.

The width and height of all particles are fixed and equal to the original template model. For both algorithms the number of bins K was set to 32 for each histogram. Three values for the

total number of samples N were tested (50, 100 and 200) to observe the variations in the computational load difference between the two tested algorithms when the maximum number of particles varies. When the maximum number of particles N decreases, we expect the computation load difference between the two algorithms to decrease. This is because the number of particles that SETA saves at each time step decreases.

The specific parameters that define SETA are set as follows: p is taken to be 10, β is set to 5, and $\zeta$is set to 0.9.

Recall that γ is a threshold on the weight of particles, and that whenever the number of particles at least as good as that threshold (with weights lower or equal) reaches β , sampling is interrupted and the rest of the particle list is effectively pruned. Also recall that δ is another threshold such that whenever the best particle has a weight at least as good as that threshold, the template model is updated.

The particle weights are related to the dimension of their particle area (number of pixels). Instead of normalizing the weights, which would require a division, we multiply the threshold values for δ and γ by the particle area A, as shown in (6.7) and (6.8).

$$\delta = 0.6 \times A, \tag{6.7}$$

$$\gamma = 1 \times A, \tag{6.8}$$

6.5 Results and Discussions

To compare the performance of SETA with respect to CONDENSATIONh, both algorithms were implemented and evaluated on a 3.4 GHz i7 processor using Matlab. We used a large dataset that consists of 19 video sequences from which we selected six sequences representative of the full set to illustrate the impact of various settings, environments, tracking subjects, behaviors, target speed, as well as static and moving cameras.

6.5.1 Evaluation methodology

The first sequence was extracted from the first video in the PETS 2001 dataset (Computational Vision Group, 2011). This video shows an outdoor parking scene where people and cars are passing in the camera field of view. This scene is characterized by a highly cluttered background. Given that both tested algorithms are designed for tracking a single target, we isolated one target in the scene. The video sequence starts and ends when the target enters and exits

the field of view of the camera. The sequence (Sequence 1) shows a pedestrian crossing a lawn. Sequence 2 is taken from the Visor repository (The Imagelab Laboratory Of University Of Modena And Reggio Emillia, 2011). The scene takes place indoors and shows two persons that are moving between several occluding objects of different sizes. The person that stays the longest in the scene is chosen as target. Sequence 3 consists of a black mouse in a transparent cage. The video is taken from (MIT CSAIL Computer Vision Research Group, 2011). This video was chosen to expand on tracking subjects and backgrounds. The first three sequences were captured with a static camera while the others were captured with a moving camera. A moving camera presents further challenge due to a continuous change in background. Sequence 4 is taken from the UCF dataset (Center For Research In Computer Vision (Ucf), 2011). The sequence represents a diving athlete. The person in each frame rotates several times before hitting the water. This presents a continuous change of the target appearance and in particular its intensity distribution. The two remaining sequences are cuts from YouTube videos. Each sequence lasts as long as the target is in the field of view of the camera. Sequence 5 is a cut from a soccer game (Ronald Brown, 2011). Sequence 6 is a cut from a hockey game video (HockeyWebCaster, 2011). Targets in sequences 5 and 6 are continuously changing speed and scale, and are viewed from different angles. In addition, the target is surrounded by similar objects (players) and moving with a relatively high speed. The targets are also subject to occasional partial or total occlusion caused by other objects. The targets are mainly chosen as one of the players that remain longest in the camera view field. This is usually the player with the ball or the puck. Table 6.1 summarizes the tested videos, and Figure 6-6 shows one frame for each sequence with a square indicating the tracked target.

Table 6.1 Test sequences

Sequence #	Frames	Scene	Camera
1	416	Outdoor parking	Static
2	553	Indoors	Static
3	873	Mouse in cage	Static
4	55	Diver	Moving
5	401	Soccer game	Moving
6	134	Hockey game	Moving

We calculated two metrics to evaluate our algorithm: execution time and tracking error (err). The tracking error (6.9) is calculated as the normalized distance between the center of the target position calculated by the tracking algorithm (x_t, y_t) and the center of the respective ground truth target (x_m, y_m). The ground truth is obtained manually by the authors on the frames considered for validation. The execution time is the CPU time that is required by the

algorithm to process the given sequence.

$$\text{err} = \sqrt{\left[\left(\frac{x_t - x_m}{\text{target width}}\right)^2 + \left(\frac{y_t - y_m}{\text{target height}}\right)^2\right]}, \quad (6.9)$$

To check the number of computations that SETA saves, we kept track of two measures, $\beta count$, which is the average number of times that the number of particles calculated does not reach N for each frame, and $\gamma count$ which is the average number of particles calculated before the β particles are found.

The values of those metrics were calculated for ten executions for each of the two algorithms on every sequence for each N. The means of $\beta count$ and $\gamma count$ were calculated for each frame and for every execution. The execution time and tracking error were evaluated for sixty random frames and for every execution. For the sequence with fewer than sixty frames, the whole sequence was considered. Finally, the mean of each metric was calculated. The duration of each execution was measured in seconds.

6.5.2 Experimental accuracy and acceleration results

Figure 6.3 shows the mean results for the tracking error for N = 50, 100 and 200. For the three values of N, the results show that SETA has a precision similar to CONDENSATIONh (CFh) for each sequence. The purpose of SETA is not to improve the tracking accuracy of the CONDENSATION algorithm, but rather to reduce its computational complexity and achieve a more efficient implementation. Consequently, achieving equivalent tracking accuracy to CONDENSATIONh is satisfactory.

Figure 6.3 also shows that in the considered video sequences, when increasing the number of

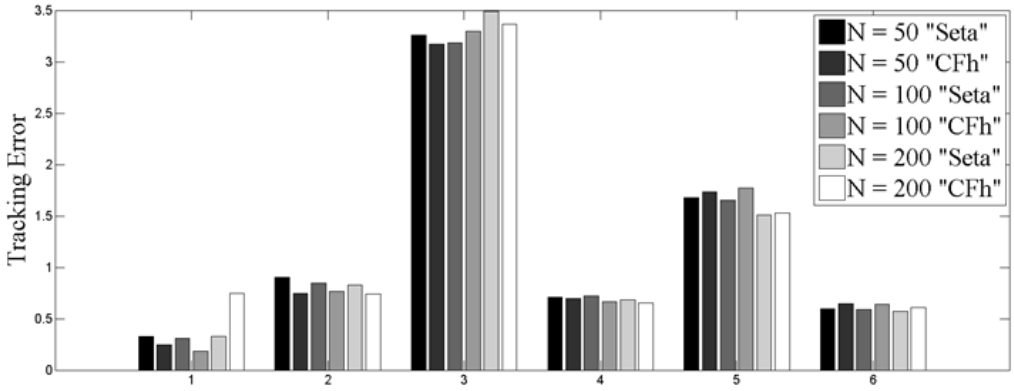


Figure 6.3 Tracking error (see equation 9) for the SETA and CONDENSATIONh (CFh).

particles N with the CONDENSATIONh algorithm, the tracking accuracy is not improved. In some cases, as for sequence 1 and sequence 2, the tracking accuracy actually deteriorates when N increases. This indicates that an increase in the number of particles was not necessary to improve the tracking accuracy. However, it is not always possible to determine the most efficient number of particles N that satisfies the best compromise between computational efficiency and tracking accuracy and the optimal value of N is certainly not known a priori. Thus, an algorithm such as SETA that dynamically adjusts the number of particles can be highly beneficial. Finding the best means of performing dynamic adjustment of N as a function of the context, system objectives and specifications remains an area for further research.

The time acceleration achieved by SETA over CONDENSATIONh is calculated as follows:

$$acceleration = \frac{time_{CFh}}{time_{SETA}}, \tag{6.10}$$

The acceleration is shown in Table 6.2 for the same three values of N. The acceleration is at least 3.81×, but can be as high as 58.9 ×. For sequences 1, 2, 5 and 6, there is a significant difference in acceleration as the number of particles increases to N = 200.

Table 6.2 SETA acceleration over CFh

Sequence #	**Acceleration**		
	N = 50	**N = 100**	**N = 200**
1	3.81 ×	3.87 ×	55.7 ×
2	7.79 ×	12.1 ×	58.9 ×
3	2.80 ×	3.70 ×	3.70 ×
4	3.03 ×	4.35 ×	6.38 ×
5	4.49 ×	7.89 ×	39.5 ×
6	4.72 ×	7.27 ×	13.9 ×

Table 6.3 illustrates the reduction in average complexity achieved by SETA for N = 100. The column $\beta count$ summarizes the average number of frames where the number of particles computed did not reach N. The column $\gamma count$ summarizes the average number of particles computed out of the total number of particles N when β is reached. The last column, saved comp, gives an estimate of the reduction in the number of computations in percent. This reduction is estimated from the number of samples processed with SETA as compared to CONDENSATIONh. The saved comp estimate is calculated as follows:

$$saved\ comp = \frac{[(nb\ frames - \beta count) \times N + \beta count \times \gamma count] \times 100}{(nb\ frames \times N)}. \tag{6.11}$$

Comparing the saved comp column in Table 6.3 and the N=100 column in Table 6.2 shows

that the saved computations calculated are consistent with the measured acceleration.

6.5.3 Acceleration analysis

The computation performance can also be assessed by considering the type and frequency of occurrence of operations used by each algorithm. A detailed list of significant operations for each algorithm is shown in Table 6.4 and Table 6.5. The expressions in Table 6.5 explain why the mean value registered for the number of saved computation in Table 6.3 is ~85%. Thus, we assume that on average $N/5$ particles are computed in each cycle with SETA.

Table 6.3 Average Computation calculation for N = 100

Sequence #	βcount	γcount	saved comp (%)
1	415	10.2	89.6
2	552	6.18	93.7
3	650	8.39	68.2
4	47.6	7.18	80.3
5	400	7.10	92.6
6	130	5.41	91.5

Table 6.4 CONDENSATIONh significant operations

Step	Operations	
Sampling	$N \times random\ number\ generation$	
Prediction	$N \times addition$ $N \times random\ number\ generation$	
Measurement	Weight calculation	$N \times histogram\ generation$ $N \times addition$ $N \times subtraction$ $N \times absolute\ values$
	Normalization and sorting	$N \times division$ $N \times addition$ Sorting of N particles

Table 6.6 compares the number of operations required by the two algorithms. In general, $p \ll N$. Recall that in this implementation, we used p = 10 and N = 50, 100 and 200. The sorting algorithm used is the Quick Sort implementation of Matlab. It is of the order of $O(n\ log\ n)$. Clearly, the number of operations used by SETA is significantly reduced.

The SETA algorithm described in Figure 6.2 is a framework that can be used in different applications and with different observation models. The intensity histogram used for this application is by far the most computationally demanding operation in the algorithm and it dominates all the other operations. For example, computing a histogram for a rectangle of modest

size enclosing a particle, e.g. 230 × 240 pixels, takes approximately 8 M clock cycles. Consequently, reducing the number of particles and thus the number of histograms computed to one quarter explains the potentially large computational requirements advantage achieved by SETA. However, when used in another application and with other observation models, such as radar tracking, other modifications that SETA brought on can have further significant impact when implemented on a hardware platform.

Table 6.5 SETA significant operations

Step	**Operations**	
Sampling	Sorting of $N/5$ particles $1 \times division$ $(P-2) \times multiplication$ $(P-1) \times subraction$	
Prediction	$N/5 \times addition$ $N/5 \times random\ number\ generation$	
Measurement	Weight calculation	$N/5 \times histogram\ generation$ $N/5 \times addition$ $N/5 \times subtraction$ $N/5 \times absolute\ values$

Table 6.6 Operation Comparison

Operations	**Number of particles involved**	
	CONDENSATIONh	SETA
Uniform random number generation	N	0
Gaussian random number generation	N	$N/5$
Histogram calculation	N	$N/5$
Addition	$3 \times N$	$2 \times N/5$
Subtraction	N	$p + \frac{N}{5} - 1$
Multiplication	0	$p-2$
Division	N	1
Sorting	N	$N/5$

6.5.4 Experimental particle variance analysis

In general, a particle filter requires a high variance in the position of its particles. The high variance is crucial to recover from occlusion errors and to maintain tracking when abrupt motion takes place. For this reason, we measured and compared the particle position variance produced by both algorithms.

Table 6.7 summarizes the mean results for the position variance calculation in the x and y dimensions, respectively. SETA has a smaller variance than the CONDENSATIONh algorithm. The smaller variance is due to the fact that the best particles are chosen to generate the

particles of the next generation. This causes the particles to spread on a smaller region with respect to the region covered by the particles generated by the CONDENSATIONh algorithm. However this property seems to benefit our algorithm in the cases where there are distractions (near objects with similar color distribution) without undermining its accuracy in the cases of partial occlusions.

Table 6.7 Mean Variance Calculation

		N = 50		N = 100		N = 200	
		SETA	CFh	SETA	CFh	SETA	CFh
sequence 1	X	195	185	197	190	195	187
	Y	208	266	204	271	209	273
sequence 2	X	149E2	149E2	152E2	150E2	154E2	151E2
	Y	293E1	309	292E1	309E1	291E1	308E1
sequence 3	X	209.5	213.28	213	217	214	236
	Y	1.18	6.12	1.18	5.92	1.18	7.52
sequence 4	X	218.2	103E1	217	105E1	204	111E1
	Y	206.4	106E1	217	108E1	210	111E1
sequence 5	X	218 E1	196E1	214E1	202E1	216E1	199E1
	Y	214 E2	199E1	222E1	205E1	218E1	210E1
sequence 6	X	22	88.6	21.5	91.7	21.6	91
	Y	23.3	126	21.6	129	21.7	122

To add to the previous variance analysis, Figure 6.4 and Figure 6.5 show plots of the sample position variance of SETA in the x and y dimensions, respectively. For each subsequence, and for N = 100, we picked 100 random frames and calculated the variance value in the x and the y dimensions. The average of those values was also calculated. Finally, all the numbers were normalized and plotted in chronological order. Figure 6.4 and Figure 6.5 show that SETA maintained a variance that fluctuated around the average. The variance did not degenerate with time.

6.5.5 Comparison with other acceleration methods

Other authors have attempted to improve the CONDENSATION algorithm and published their results. Table 6.8 displays the mean results reported by those papers and this work for comparison.

The Acceleration column displays the accelerations achieved by each algorithm over the CONDENSATION algorithm. For the SETA, we calculated the average of the results shown in Table 6.2 for each N.

Maggio and Cavallaro (2005) combined the particle filter with a mean shift. The algorithm was tested on three videos representing a highway scene, a table tennis scene and a soccer

scene. The authors reported that their algorithm is 32% faster than the standard algorithm. Using the mean shift allowed them to reduce the number of particles from 150 that they used for the standard CONDENSATION, to 30 particles that they used for their algorithm. These results are consistent with our findings. Being able to decrease the number of particles contributes to decreasing the processing time of the algorithm.

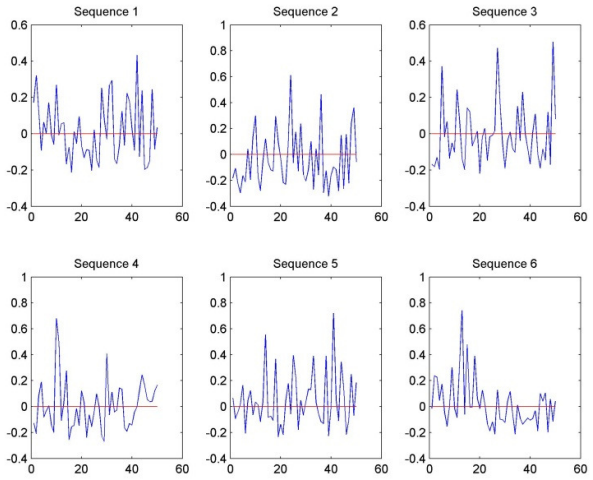


Figure 6.4 The variance of each sequence in the x dimension for N = 100.

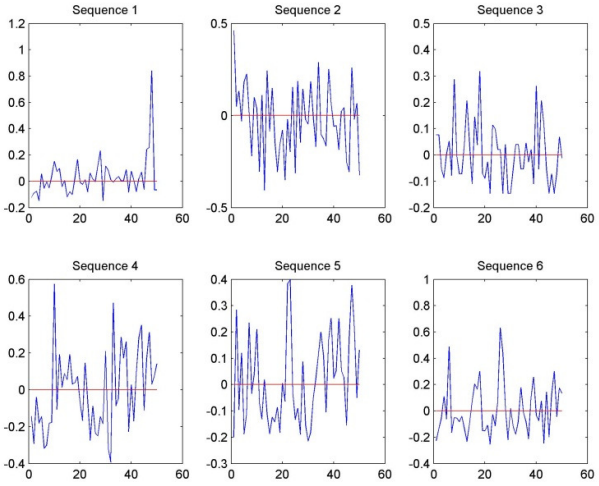


Figure 6.5 The variance of each sequence in the y dimension for N = 100.

Deautscher et al. (2000) also decreased the number of particles from 4000 to 400 by coupling the CONDENSATION to a simulated annealing algorithm. They tested their algorithm on three sequences of a person doing two activities. The first activity represents a person turning in a circle. The second activity represents a person stepping over a box, turning around and then stepping over the box again. These authors obtained reduced computational complexity.

They reported a processing time decrease by a factor of 4. SETA achieved approximatly the same acceleration for 50 particles.

Li et al. (2003) combined the CONDENSATION algorithm with the Kalman Filter (KPF) and the Unscented Kalman Filter (UPF). They tested the two algorithms on a sequence of a hand in a pointing gesture. Both algorithms reduced the number of particles while improving accuracy. However, the large decrease in the number of particles did not necessarly decrease their computation time. Indeed, the Unscented Kalman filter added computationally demanding operations to the CONDENSATION algorithm. This greatly increased the processing time from 67 ms to 460 ms and decreased.

Table 6.8 Acceleration Comparison

Algorithm	Acceleration
Maggio & Cavallaro (2005), ($N_{(M\&G)}$ = 30, $N_{CF} = 150$)	1.47
Deutscher et al. (2000), ($N_{(D.et\ al.)}$ = 400, $N_{CF} = 4000$)	4
Li et. al. (2003), (N_{KPF} = 50, $N_{CF} = 500$)	1.21
Li et. al. (2003), (N_{UPF} = 20, $N_{CF} = 500$)	0.15
SETA (N = 50)	4.44
SETA (N = 100)	6.53
SETA (N = 200)	29.7

Figure 6-6 Snapshots from the processed sequences.

To summarize, when considering the acceleration that we obtained with SETA, it compares favorably with the other reported improved particle filters while maintaning the same accuracy as the CONDENSATION algorithm.

However we should keep in mind that when attempting to provide a hardware implementation

of any of those algorithms, one would be faced with the same shortcomings as when implementing a particle filter. For instance, both the Unscented Kalman filter and the simulated annealing algorithms require at least one random value generation per particle at every iteration, while the mean shift tracker requires at least one division per particle for every iteration to compute the Bhattacharyya coefficient.

6.6 Conclusion

In this paper, we proposed a new algorithm inspired by the CONDENSATION algorithm. In our algorithm, several simplifications were proposed to facilitate future hardware implementations. We also proposed a means to dynamically reduce the average number of particles effectively processed at each time step, while keeping the same maximum number of particles as the reference algorithm. These simplifications were implemented in software and it was shown that our algorithm maintains roughly the same accuracy. A detailed complexity analysis showed that interrupting sampling reduces the average number of processed particles to a fifth of that required in CONDENSATION. In addition, SETA achieved a mean acceleration ranging between 4.4x and 12x and a mean reduction factor of the number of computed particles by a value of 85% for a number of samples N = 100. As shown by the results, the computation reduction did not undermine the robustness of the tracker and the particle position variance did not degenerate with time.

CHAPTER 7 GENERAL DISCUSSION

Initially, we intended to use the particle filter algorithm to track rodents under the settings imposed by typical biomedical environments. This choice was justified by the popularity of the particle filter and by recent work where the particle filter was used to track rodents. As mentioned in Chapter 2, Pistori et al. (2010) and Goncalves et al. (2007) used a combination of the particle filter and the k-means algorithm to track rodents. Brason and Belongie (2005) also used a particle filter to track several mice in a cage.

However, when we tried to use the particle filter on our videos, we faced several problems. For the particle filter to achieve accurate tracking results, the number of particles should be large. Also, the observation model should be robust and appropriate to represent and distinguish the properties of the target.

Rodents' bodies undergo excessive deformations when they move. These deformations affect the shape, dimension, texture and color of the rodent. When considering using the particle filter as a stand-alone algorithm to track rodents, these excessive deformations make it impractical to find an efficient observation model. While Pistori et al. (2010), Goncalves et al. (2007), and Brason and Belongie (2005) all succeeded in finding suitable observation models, the tracking in those cases was preceded by a segmentation step that isolated the targets. Thus, simpler observation models could be used. Moreover, as discussed in Chapter 4, segmentation was not appropriate when treating our videos.

We attempted to use the particle filter as part of the algorithm described in Chapter 4 as a means to coarsely track the rodent. The observation model that we used was a rectangular window in which we computed the three features used in the method described in Chapter 4. However, the sliding window technique returned better results. This is because the sliding window technique scans the ROI exhaustively and covers all the possible locations for the target. It was possible to increase the number of particles to increase the coverage of the ROI. This would return similar results as the sliding window technique. It would also require the extra computation necessary to the generation of each particle (resampling phase, propagation phase, etc.). Moreover, the second phase of the algorithm requires accurate tracking in the first phase to lock on the correct edglets. A small offset in the tracking phase may increase the possibility, in the second phase, of locking the window boundaries on edglets that do not belong to the target.

Even though the particle filter was not adequate in this case, it is still a widely used algorithm in rodent tracking. For this reason, we suggested several contributions to accelerate and simplify the algorithm.

In the first article (Chapter 4), a rectangular box is used to extract the rodent at each frame. It seemed logical to use this box for feature extraction to detect and recognise the three behaviors targeted in the second article (Chapter 5). We considered three features: the height of the box, the height-to-width ratio, and the direction of motion. The direction of motion in this case is defined as the direction of change between the heights of two consecutive boxes. For instance, if the height of the box at time t is greater than the height of the box at time $t + 1$, then the direction of motion is negative and the rodent is going down. These two features were used to detect the posture of the rodent (on two feet or on four feet). The results for the posture detection accuracy were poor. This is because the input (the dimensions of the box) was noisy.

Consequently, we calculated the same features on ten consecutive frames and considered the average. The accuracy improved, but it was still not satisfactory for two reasons; First, the calculated dimensions of the box were noisy. Second, the discrimination ability of the chosen features is limited. For instance, at some given height range, the rodent has the same probability of being on two feet or being on four feet. Moreover, calculating the position of the rodent at each frame is not essential to detect and identify the three behaviors. For these reasons, the rectangular box model was dropped and the MHI model was considered. As mentioned in Chapter 5, the MHI was used for action recognition (Yau, W., et al., 2006; Ahad, Md A. R., et al, 2012; Bobick and Davis; 1996). We found that MHIs present richer information than a simple rectangular box. They are also more adapted for rodent action recognition, as described in Chapter 5.

Tracking the position of the rodent is still an important task. As mentioned in the introduction (Chapter 1), tracking the rodent and determining its motion pattern is useful to infer its mood and psychological state (Ishii, H., et al., 2007).

CHAPTER 8 CONCLUSION

8.1 Summary of the work

In this thesis, we proposed solutions to practical computer vision problems in rodent tracking and monitoring. We exerted minimal constraints over illumination, cage localization and background settings. Typical transparent cages were used to enable visibility. The contents of the cages were undisturbed. The only required constraint is that only one rodent be allowed in a cage at one time.

The first proposed solution tracks and extracts a rodent under the outlined conditions. The solution is based on a model to represent the rodent that consists of three features calculated in a rectangular window that bounds the rodent. The three features are HOG, OHI and the absence of motion. To determine the position of the rodent in each frame, a sliding window technique is used to calculate its most probable position. The solution is also based on a scheme to extract the rodent. This scheme fits the boundaries of the tracking window to the contour of the rodent using the edglets of the target and a system of vertical and horizontal pulses.

There are two contributions for this solution. The first one is the introduction of the new feature OHI. This feature extends HOG by using the intensities of the frame instead of the gradients. The second contribution consists of introducing a new segmentation method that uses an e-background subtraction and the edglets of the target to fit the boundaries of the rectangular window to the contour of the rodent.

This method was tested on rat videos recorded under typical biomedical environments. It outperformed a state of the art tracking method and its performance was consistent when tested on different backgrounds and rat sizes.

The second proposed solution detects and distinguishes three behaviors (exploring, rearing and being static) in rodents under the outlined conditions. The solution includes a rule-based method combined with an MSC based on features extracted from the MHI computed in each frame.

The solution brought two contributions. The first one is a new method to detect the three behaviors based on MHI. The second contribution is a new fusion rule that combines the estimation of several SVMs. This fusion rule is proposed due to its simplicity and its ability to favor the classifier with the strongest estimation at each call.

The method was tested under typical biomedical laboratory conditions and it achieved a recognition accuracy of 87%. This accuracy is sufficient with respect to what is acceptable in animal monitoring in biomedical laboratories (Nie, Y., et al., 2011). It compares favorably to state of the art methods.

The third related solution proposes a tracking algorithm that simplifies the CONDENSATION algorithm while maintaining its apparent functionalities and its accuracy. The simplified tracking algorithm reduces the computation load and the number of complex operations with respect to the original algorithm. This may favor an efficient hardware implementation in the future.

This solution contributed in a new method for sampling that reduced the average number of particles generated at each time step. It also contributed in a simplified method to assign the number of particles at a certain time step generated from the particles at the previous time step.

The proposed solution was implemented in software, and it was shown that it maintains roughly the same accuracy as the CONDENSATION algorithm. It was also shown that the solution reduces the average number of generated particles at each time step to one fifth, while keeping the maximum number of particles the same as in the original algorithm. SETA achieves a mean acceleration ranging between 4.4× and 12× and a mean reduction factor of the number of computed particles by a value of 85% for a number of samples N = 100.

8.2 Future work

Even though the work described in this thesis presented multiple contributions in the field of rodent tracking using computer vision techniques, several improvements could be proposed to the solutions presented. Moreover, many extensions could be proposed.

For instance, rodent tracking may be improved by exploring other features to include or replace the ones used to track the rodent. For instance, one may also include a feature based on

color, given that this dimension is not explored in the proposed algorithm. The color channels include information that is not covered by the used features.

The segmentation of the rodent may be extended to extract the exact contour of the rodent instead of the bounding box. One should investigate using the nearest edglets to the bounding box. Extracting the exact contour of the rodent would permit to locate the positions of its different body parts. This information may be useful to determine certain behaviors. For instance, when repetitive motion takes place in the head region, this may indicated grooming.

For the rodent action recognition aspect, it is clear that much further work could be done. The thesis only covers three basic actions: rearing, exploring and being static. The work may be extended to include other basic actions such as grooming, scratching or climbing. Other more complex action such as undergoing seizures may also be included. MHI may be combined with other features and information to detect some of these other behaviors.

Finally, our work is limited to one rodent in a cage. This work may be extended to include several rodents in one cage. This will present extra challenges for tracking as when the rodent occlude one another, touch, or cross each other when going in different directions. Having several rodents in one cage will also extend the range of behaviors to detect to include interaction behaviors.

REFERENCES

AHAD, MD.A.R., TAN, J. K., KIM, H. and ISHIKAWA, S. (2010). Action Recognition by Employing Combined Directional Motion History and Energy Images. *Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 73-78).

AHAD, MD.A.R., TAN, J.K., KIM, H. and ISHIKAWA, S. (2012). Motion history Image: Its variants and applications. *Machine Vision and Applications (MVA)*, 23(2), 255-281.

AMANO, S., YOKOVAMA, M., TORRI, R., TANAKA, K., IHARA, N. and HAZAMA, F. (1997). High performance seizure-monitoring system using a vibration sensor and videotape recording: behavioral analysis of genetically epileptic rats. *Laboratory Animal Science*, 47(3), 317-320.

American Association for Laboratory Animal Science. *Use of Animals in Biomedical Research. Understanding the Issues.* Consulted on the 23rd of January 2013, taken from: http://www.aalas.org/pdf/08-00007.pdf.

AVIDAN, S. (2007). Ensemble Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2), 261-271.

BAKER, S. and MATTHEWS, I. (2001). Equivalence and efficiency of image alignment algorithms. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1, 1090–1097.

BAUMBERG, A. and HOGG, D. (1994). Learning flexible models from image sequences. *European Conference on Computer Vision (ECCV)* (pp. 299–308).

BELONGIE, S., BRANSON, K., DOLLAR, P. and RABAUD, V. (2005). Monitoring animal behavior in the smart vivarium. *Measuring Behavior* (pp. 70-72).

BEN-ARIE, J., WANG, Z., PANDIT, P., RAJARAM, S. (2002). Human activity recognition using multidimensional indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24(8), pp.1091, 1104.

BISSACCO, A., CHIUSO, A., MA, Y., SOATTO, S. (2001). Recognition of human gaits. *Conference on Computer Vision and Pattern Recognition (CVPR)* (vol. 2, pp. II-52–II-57).

BLACK, M. J. and JEPSON, A. D. (1998). A probabilistic framework for matching temporal trajectories condensation based recognition of gestures and expressions. *European Conference on Computer Vision (ECCV)* (pp. 909-924).

BLAKE, A., CURWEN, R., and ZISSERMAN, A. (1993). A framework for spatiotemporal control in the tracking of visual contours. *International Journal on Computer Vision*, vol. 11(2), pp. 127-145.

BOBICK, A. and DAVIS, J. (1996). Real-time recognition of activity using temporal templates. *Workshop on the Applications of Computer Vision (WACV)* (pp. 38-42).

BOBICK, A. and DAVIS, J. (2001). The Recognition of Human Movement Using Temporal Templates. *Pattern Analysis and Machine Intelligence (PAMI)*, 23(3), 257-267.

BRADSKI, G. R. (1998). Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, Q2.

BRANSON, K. and BELONGIE, S. (2005). Tracking Multiple Mouse Contours (without Too Many Samples). *Computer Vision Pattern Recognition (CVPR)* (vol. 1 pp. 451-457).

BRETZNER, L., LAPTEV, I., and LINDEBERG, T. (2002). Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. *Fifth IEEE International Conference on Automatic Face and Gesture Recognition* (pp423-428).

BURGOS-ARTIZZU, X. P., DOLLAR, P., LIN, D., ANDERSON, D. J., PERONA, P. (2012). Social behavior recognition in continuous video. *Computer Vision Pattern Recognition (CVPR)* (pp. 1322-1329).

CANADIAN FOR HEALTH RESEARCH. *Rats and Research.* Consulted on the 23rd of January 2013, taken from http://www.chrcrm.org/en/rats-and-research.

CANNY, J. (1986). A computational approach to edge detection. *Pattern Analysis and Machine Intelligence (PAMI)*, 8, 679-698.

CENTER FOR RESEARCH IN COMPUTER VISION (UCF). *UCF50*. Consulted in 2012, taken from: http://server.cs.ucf.edu/vision/data.html.

CHELLAPPA, R. and ZHOU, S. (2003). *Face Tracking and Recognition from Video*. Handbook of Face Recognition. *Springer*.

CLEVERSYS. *Home Cage Environement*. Consulted on February 2013, Taken from: http://www.cleversysinc.com/products/hardware/home-cage-environment.

COMANICIU, D. and MEER, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24, 603-619.

COMANICIU, D., RAMESH, V. and MEER, P. (2003). Kernel-Based Object Tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence (PAMI)*, 25(5), 564-577.

COMANICIU, D., RAMESH, V., and MEER, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol.25(5), pp. 564-577.

COMPUTATIONAL VISION GROUP. *PETS: Performance Evaluation of Tracking and Surveillance*. Consulted in 2013, taken from: http://www.cvg.rdg.ac.uk/slides/pets.html.

CUTLER, R., TURK, M. (1998). View-based interpretation of real-time optical flow for gesture recognition. *International Conference on Automatic Face and Gesture Recognition* (pp. 416–421).

DALAL, N., TRIGGS, B. (2005). Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition (CVPR)* (vol.1, pp. 886-893).

DALAL, N., TRIGGS, B., and SCHMID, C. (2006). Human detection using oriented histograms of flow and appearance. *European Conference on Computer Vision (ECCV)* (pp. 428-441).

DALLAL, N. (2006). *Finding people in images and videos*. Institut National Polytechnique de Grenoble.

DALLAL, N. and TRIGGS, B. (2005). Histograms of Oriented Gradients for Human Detection. *Computer Vision Pattern Recognition (CVPR)* (pp. 886-893).

DAVIS, J. and BOBICK, A. (1997). The representation and recognition of human movement using temporal templates. *Computer Vision Pattern Recognition (CVPR)* (pp. 928-934).

DELLAERT, F., FOX, D., THRUN, W. S. (1999). Monte Carlo localization for mobile robots. *International Conference on Robotics and Automation* (*ICRA)* (vol. 2, pp. 1322-1328).

DEUTSCHER, J., BLAKE, A. and REID, I. (2000). Articulated body motion capture by annealed particle filtering. *Computer Vision Pattern Recognition* (*CVPR)* (vol 2, pp. 126-133).

DOLLAR, P., RABOUD, V., COTTRELL, G. and BELONGIE, S. (2005). Behavior recognition via sparse spatio-temporal features.VS-*PETS* (pp. 65-72).

DOLLAR, P., TU, Z. and BELONGIE, S. (2006). Supervised Learning of Edges and Object Boundaries. *Computer Vision Pattern Recognition* (*CVPR)* (pp. 1964-1971).

DONG, W.P., KWON, J., and LEE, K.M. (2012). Robust Visual Tracking using Autoregressive Hidden Markov Model. *Computer Vision and Pattern Recognition (CVPR)* (1964-1971).

FARAH, R., GAN, Q., LANGLOIS, J. M. P., BILODEAU, G.A., SAVARIA, Y. (2011). A Tracking Algorithm Suitable for Embedded Systems Implementation. *International Conference on Electronics, Circuits, and Systems (ICECS)* (pp. 256-259).

FARAH, R., LANGLOIS, J.M.P. and BILODEAU, G.A. (2011). RAT: Robust Animal Tracking. I*nternational Symposium on Robotic and Sensors Environment (ROSE)* (pp. 65-71).

FARAH, R., LANGLOIS, J.M.P. and BILODEAU, G.A. (2013). Catching a Rat by its edglet. *IEEE Transactions on Image Processing (TIP)*, 22(2), 668-678.

FOX, D. (2003). Adapting the Sample Size in Particle Filters Through KLD-Sampling. The International Journal on Robotics Research (IJRR), 22(12), 985-1004.

FOX, W. B., DELLAERT, F., THRUN, S. (1999). Monte Carlo Iocalization: efficient position estimation for mobile robots. *Proceedings of the Sixteenth National Conference on Artificial Intelligence* (pp. 343-349).

FREUND, Y. and SCHAPIRE, R. (1997). A Decision-theoretic Generalization of On-line Learning And an Application to Boosting. *Journal of Computer and System Sciences*, vol. 55(1), pp. 119-139

FUENTES, L.M., and VELASTIN, S.A. (2001). Foreground Segmentation using Luminance Contrast. *WSES International Conference on Speech and Image Processing* (pp. 2231–2235)

FUENTES, L.M., and VELASTIN, S.A. (2006). People tracking in surveillance applications. *Image and Vision Computing*, vol.24 (11), pp. 1165-1171.

GIBBS, S., CHATTOPADHYAYA, B., DESGENT, S., AWAD, PN., CLERK-LAMALICE, O., LEVESQUE, M., VIANNA, RM., RÉBILLARD, RM., DELSEMME, AA., HÉBERT, D., TREMBLAY, L., LEPAGE, M., DESCARRIES, L., DI, G. and CARMANT, L. (2011). Long-term consequences of a prolonged febrile seizure in a dual pathology model. *Neurobiology of Disease*, 43(2), 312-321.

GONCALVES, W.N., MONTEIRO, J.B.O., SILVA, J., MACHADO, B.B., PISTORI, H. and ODAKURA, V. (2007). Multiple Mice Tracking using a Combination of Particle Filter and K-Means. *Computer Graphics and Image*, 173-178.

GRANLUND, G., and KNUTSON, H. (1995). *Signal Processing for Computer Vision*. The Netherlands: Kluwer Academic Publishers.

GREER, J.M., CAPECCHI, M.R. (2002). Hoxb8 is required for normal grooming behavior in mice. *Neuron*, 33(1), 23-34.

GUO, Y., XU, G., TSUJI, S. (1994). Understanding human motion patterns. *International Conference on Pattern Recognition* (vol. 2, pp. 325–329).

HAAR, A. (1911). Zur Theorie der orthogonalen Funktionensysteme. *Mathematische Annalen*, vol.71(1), pp. 38-53.

HAGER, G.D. and BELHUMEUR, P.N. (1998). Efficient region tracking with

HAO, J. and DREW, M. S. (2002). A predictive contour inertia snake model for general video tracking. *The International Conference on Image Processing (ICIP)* (pp. 413-416).

HARRIS, C., and STEPHENS, M. (1988). A Combined Corner and Edge Detection. *Alvey Vision Conference* (pp. 147-151).

HASTIE, T., TIBSHIRANI, R., and FRIEDMAN, J. (2001). *The Elements of Statistical Learning*. Basel: Springer Series in Statistics, Springer.

HOCKEYWEBCASTER. *Top 10 NHL Shootout Goals 2010-11 (HD)*. Consulted in 2012, taken from: http://www.youtube.com/watch?v=Krq0SOyQkz0.

HORN, B.K.P., and SCHUNCK, B.G. (1981). Determining Optical Flow. *Artificial Intelligence*, vol. 17(1-3), pp. 185-203.

HOU, X. and ZHANG, L. (2007). Saliency Detection: a Spectral Residual Approach. *Conference on Computer Vision and Pattern Recognition* (*CVPR)* (pp. 1-8).

HU, M. (1962). Visual pattern recognition by moment invariants. I*RE Transaction on Information Theory*, 8(2), 179–187.

ISARD, M. and BLAKE, A. (1996). Contour tracking by stochastic propagation of conditional density. *European Conference on Computer Vision (ECCV)* (vol. 1064, pp. 343-356).

ISARD, M. and BLAKE, A. (1998a). CONDENSATION—Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, 29(1), 5-28.

ISARD, M. and BLAKE, A. (1998b). Icondensation: Unifying low-level and high-level tracking in a stochastic framework. *European Conference on Computer Vision (ECCV)* (vol. 1406, pp. 893-908).

ISARD, M., and BLAKE, A. (1998c). A Smoothing Filter for CONDENSATION. *European Conference on Computer Vision (ECCV)* (vol. 1406, pp. 767-778).

ISARD, M., and BLAKE, A. (1998d). A mixed-state condensation tracker with automatic model-switching. *International Conference on Computer Vision (ICCV)* (pp. 107-112)

ISARD, M., and MACCORMICK, J.(2001). BraMBLe: A Bayesian multiple-blob tracker. *International Conference on Computer Vision (ICCV)* (vol. 2, pp. 34-41).

ISHII, H., OGURA, M., KURISU, S., KOMURA, A., TAKANISHI, A., IIDA, N. and KIMURA, H. (2007). Development of autonomous experimental setup for behavior analysis of rats. *Intelligent Robots and Systems*. 4152 – 4157.

JACKSON, W.S., TALLAKSEN-GREENE, S.J., ALBIN, R.L. and DETLOFF, P.J. (2003). Nucleocytoplasmic transport signals affect the age at onset of abnormalities in knock-in mice expressing polyglutamine within an ectopic protein context. *Human Molecular Genetics*, 12(13), 1621-1629.

JANG-HEE, Y., MARK, S.N., and CHRIS, J.H. (2002). Extracting Human Gait Signatures by Body Segment Properties. *IEEE Southwest Symposium on Image Analysis and Interpretation* (pp. 35-39).

JHUANG, H., GARROTE, E., EDELMAN, N., POGGIO, T., STEELE, A. and SERRE, T. (2010a). Trainable, vision-based automated home cage behavioral phenotyping. *International Conference on Methods and Techniques in Behavioral Research*, doi:10.1145/1931344.1931377.

JHUANG, H., GARROTE, E., POGGIO, T., STEELE, A. and SERRE, T. (2010b). Vision-based automated recognition of mice home-cage behaviors. *International Conference on Pattern Recognition (ICPR).*

JHUANG, H., GARROTE, E., YU, X., KHILNANI, V., POGGIO, T., STEELE, A.D. and SERRE, T. (2010c). Automated home-cage behavioral phenotyping of mice. *Nature Communication,* doi:10.1038/ncomms1064.

JHUANG, H., GARROTE, E., YU, X., KHILNANI, V., POGGIO, T., STEELE, A.D. and SERRE, T. *Vision-Based System for Automated Mouse Behavior Recognition*. Consulted on the 8th of August 2011, takes from: http://cbcl.mit.edu/software-datasets/mouse.

JHUANG, H., SERRE, T., WOLF, L., POGGIO, T. (2007). A biologically inspired system for action recognition. *International Conference on Computer Vision* (*ICCV)* (pp. 1-8).

JOHANSSON, G. (1973). Visual perception of biological motion and a model for its analysis. *Perception & Psychophysics*, vol.14(2), pp. 201–211.

JUDD, T., EHINGER, K., DURAND, F. and TORRALBA, F. (2009). Learning to predict where humans look. *International Conference on Computer Vision (ICCV)* (pp. 2106 - 2113)

KE, Y., SUKTHANKAR, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors. *Computer Vision and Pattern Recognition (CVPR)* (vol. 2, pp. 506–513).

KRATZ, L. and NISHINO, K. (2010). Tracking with local spatio-temporal motion patterns in extremely crowded scenes. *Computer Vision and Pattern Recognition (CVPR)* (pp. 693-700).

KUNCHEVA, L. I., BEZDEK, J. C. and DUIN, R. P. (2001). Decision Templates for Multiple Classier Fusion: An Experimental Comparison. *Pattern Recognition*, 34, 299-314.

KWOK, C., FOX, D., MEILA, M. (2004). Real-time particle filters. *Proceedings of the IEEE*, (vol. 92(3), pp. 469-484).

LAPTEV, I, MARSZAŁEK, M., SCHMID, C., and ROZENFELD, B. (2008). Learning realistic human actions from movies. Computer Vision and Pattern Recognition (CVPR) (pp. 1-8).

LAPTEV, I. (2005). On Space-Time Interest Points. *International Journal of Computer Vision*, vol. 64(2/3), pp.107-123.

LEUTENEGGER, S., CHLI, M., and SIEGWART, S. (2011). Brisk: Binary robust invariant scalable keypoints. *International Conference on Computer Vision (ICCV)* (pp. 2548 - 2555)

LI, G., LI, B. and CHEN, Z. X. (2008). Implementation and optimization of Particle Filter tracking algorithm on Multi-DSPs system. *Cybernetics and Intelligent Systems* (pp.152-157).

LI, P., ZHANG, T. and PECE, A. E. C. (2003). Visual contour tracking based on particle filters. *Image and Vision Computing*, 21(1), pp.111-123.

LIANG, Y., KOBLA, V., BAIS, X., ZHANG, Y. (2007). Unified system and method for animal characterization in home cages using video analysis. U.S. Patent 7 209 588.

LOWE, D.G.(2004) Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, vol. 60(2), pp. 91–110.

LUCAS, B.D., and KANADE T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. *International Conference on Artificial Intelligence (IJCAI)* (pp. 674, 679).

LV, F., NEVATIA, R. (2006). Recognition and segmentation of 3-d human action using hmm and multi-class adaboost. *European Conference on Computer Vision* (pp. 359–372).

MAGGIO, E. and CAVALLARO, A. (2005). Hybrid Particle Filter and Mean Shift tracker with adaptive transition model. *International Conference on Acoustics, Speech, and Signal Processing* (pp. 221-224).

MAINALI, P., LAFRUIT, G., YANG, Q., GEELEN, B., VAN GOOL, L., Lauwereins, R. (2013). SIFER: Scale-Invariant Feature Detector with Error Resilience. *International Journal of Computer Vision*, 10.1007/s11263-013-0622-3

MARTIN, D. R., FOWLKES, C. C. and MALIK, J. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transaction on Pattern Analysis and Machine Intelligence PAMI*, 26(5), 530-549.

MEDEIROS, H., GAO, X. and PARK, J. (2008). A parallel implementation of color-based particle filter for object tracking. *Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 1-8).

MEDEIROS, H., HOLGUIN, G. and PARK, J. J. (2010) A parallel histogram-based particle filter for object tracking on SIMD-based smart cameras. *Computer Vision and Image Understanding*, 114(11), 1264-1272.

MEIER, E. B., ADE, F. (1999). Using the condensation algorithm to implement tracking for mobile robot. *1999 Eurobot* (pp. 73-80).

MENAHEM, E., ROKACH, L. and ELOVICI, Y. (2009). Troika - An Improved Stacking Schema for Classification Tasks. *Information Sciences*, 4097-4122.

METRIS. *Labors Product Information.* Consulted on the 25th of January 2013, taken from http://www.metris.nl/en/products/laboras/laboras_information.

MIKOLAJCZYK, K., SCHMID, C. (2005). A performance evaluation of local descriptors. *IEEE Transaction on Pattern Analysis and Machine Intelligence PAMI*, vol. 27(10), pp. 1615–1630.

MIT CSAIL COMPUTER VISION RESEARCH GROUP. *Vision-Based System for Automated Mouse Behavior Recognition*. Consulted in 2011, taken from: http://cbcl.mit.edu/softwaredatasets/mouse.

MORAVEC, H.P. (1979). Visual mapping by a robot rover. *International Joint Conference on Artificial Intelligence* (vol.1, pp. 599-601).

New Behavior. *IntelliCage*. Consulted on the 25th of January 2013, taken from http://www.newbehavior.com/products/ic.

NIE, Y., ISHII, I., YAMAMOTO, K. and MATSUDA, K. H. (2009). Real-time scratching behavior quantification system for laboratory mice using high-speed vision. *Journal of real-time image processing*, 4, 181-190.

NIE, Y., ISHII, I., YAMAMOTO, K., TAKAKI, T., ORITO, K. and MATSUDA, H. (2008). High-speed video analysis of laboratory rats behaviors in forced swim test. *Automation Science and Engineering*, 206-211.

NIE, Y., TAKAKI, T., ISHII, H. and MATSUDA, H. (2011). Behavior Recognition in Laboratory Mice Using HFR Video Analysis. *IEEE International Conference on Robotics and Automation*, 1595-1600.

NOLDUS. *Computer, video, and data acquisition systems*. Consulted on the 8th of August 2011(b), taken from: http://www.noldus.com/animal-behavior-research/accessories/computer-video-and-daq-systems.

NOLDUS. *EthoVision XT*. Consulted on the 8th of August 2011, taken from: http://www.noldus.com/animal-behavior-research/products/ethovision-xt.

NUMMIARO, K., KOLLER-MEIER, E., GOOL, L. (2003). An adaptive color-based particle filter. *Image and Vision Computing*, 21(1), 99-110.

OKUMA, K., TALEGHANI, A., FREITAS, N. D. and LOWE, J.J. D. G. (2004). A Boosted Particle Filter: Multitarget Detection and Tracking. *European Conference on Computer Vision (ECCV)* (vol. 3021, pp. 28-39).

parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 20(10), 1025–1039.

PARK, S., AGGARWAL, J.K. (2003). Recognition of two-person interactions using a hierarchical bayesian network. *ACM SIGMM International Workshop on Video Surveillance* (pp. 65–76).

PEURSUM, P. WEST, G., VENKATESH, S. (2005). Combining image regions and human activity for indirect object recognition in indoor wide-angle views. *International Conference on Computer Vision* (vol. 1, pp. 82–89).

PHILOMIN, V., DURAISWAMI, R. and DAVIS, L. (2000). Quasi-Random Sampling for Condensation. *European Conference on Computer Vision (ECCV) LNCS* (vol. 1843, pp. 134-149).

PISTORI, H., ODAKURA, V., MONTEIRO, J.B.O., GONÇALVES, W.N., ROEL, A. J., and MACHADO, B.B. (2010). Mice and larvae tracking using a particle filter with an auto-adjustable observation model. *Pattern Recognition Letters*, 31, 337–346.

POLANA, R., NELSON, R. (1992). Recognition of motion from temporal texture. *Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 129–134).

QUINN, L., STEANN, T., TRAIL, B., DUXON, M., STRATTON, S., BILLINTON, A. and UPTON, N. (2003). LABORAS: Initial pharmacological validation of a system allowing continuous monitoring of laboratory rodent behavior. *Journal of Neuroscience Methods*, 130, 83-92.

RAHMAN, A. F. R. and FAIRHURST, M. C. (2003). Multiple classifier decision combination strategies for character recognition: A review. International Journal on Document Analysis and Recognition (IJDAR), 5, 166-194.

RONALD BROWN. *Best Goals of the Decade - Soccer/Football Mix HD*. Consulted in 2012, taken from: http://www.youtube.com/watch?v=to3wrQkLtZg.

RUDENKO, O., TKACH, V., BEREZIN, V. and BOCK, E. (2009). Detection of early behavioral markers of Huntington's disease in R6/2 mice employing an automated social home cage. *Behavioural Brain Research*, 203(2), 188-199.

SABOUNE, J. and LAGANIERE, R. (2009). People detection and tracking using the Explorative Particle Filtering. *International Conference on Computer Vision Workshop (ICCVW)* (pp. 1298-1305).

SCOVANNER, P., ALI, S., SHAH, M. (2007). A 3-dimensional SIFT descriptor and its application to action recognition. *ACM International Conference on Multimedia* (pp. 357–360).

SHI, J. and TOMASI, C. (1994). Good Features to Track. *Computer Vision and Pattern Recognition (CVPR)* (pp. 593-600).

SHUM, H.-Y. and SZELISKI, R. (2000). Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision,* 16(1), 63–84.

SMITH, P., SINCLAIR, D., CIPOLLA, R., WOOD, K.(1998). Effective Corner Matching. *British Machine Vision Conference, Southampton* (pp. 545-556).

STAUFFER, C. and GRIMSON, W.E.L. (1999). Adaptive Background Mixture Models for Real-Time Tracking. *Computer Vision and Pattern Recognition (CVPR)* (pp. 246-252).

THE IMAGELAB LABORATORY OF UNIVERSITY OF MODENA AND REGGIO EMILLIA. *Visor*. Consulted in 2011, taken fom: http://www.openvisor.org.

THERRIEN, C. (1989). Decision Estimation and Classification. New York: John Wiley and Sons, Inc.

TORABI, A., BILODEAU, G.-A., LEVESQUE, M., LANGLOIS, J.M.P., LEMA, P., CARMANT, L. (2008). Measuring an Animal Body Temperature in Thermographic Video Using Particle Filter Tracking. *International Symposium on Visual Computing (ISVC)* (p.p. 1081-1091).

TORABI, A., BILODEAU, G.-A., LEVESQUE, M., LANGLOIS, J.M.P., LEMA, P., CARMANT, L. (2008). Measuring an Animal Body Temperature in Thermographic Video Using Particle Filter Tracking. *4th International Symposium on Visual Computing. International Symposium on Visual Computing (ISVC)* (vol. 1, pp. 1081-1091).

TRUCCO, E., and PLAKAS, K. (2006). Video Tracking: A Concise Survey. IEEE *Journal of Oceanic Engineering*, vol. 31(2), 520-529.

TRUCCO, E., VERRI, A. (1998). Introductory Techniques for 3-D Computer Vision. Englewood Cliffs, NJ: Prentice-Hall.

TU, Z., (2005). Probabilistic Boosting-Tree: Learning Discriminative Models for Classification, Recognition, and Clustering. *International Conference on Computer Vision (ICCV)* (vol. 2, pp. 1589-1596).

UIN, R.P.W. and TAX, D.M.J. (2000). Experiments with classifiers combination rules. In J. Kittler, F. Roli (dir), *Lecture Notes in Computer Science* (vol. 1857 pp. 16-29). Berlin: Springer.

VASWANI, N., RATHI, Y., YEZZI, A. and TANNENBAUM, A. (2010). PF-MT with an Interpolation Effective Basis for Tracking Local Contour Deformations. *IEEE Transactions on Image Processing (TIP)*, 19(4), 841-857.

VELMURUGAN, R., SUBRAMANIAN, S., CEVHER, V., MCCLELLAN, J.H. and ANDERSON, D.V. (2007). Mixed-mode Implementation of Particle Filter. *Pacific Rim International Conference (PacRim)* (pp. 617-620).

VERMAAK, J., DOUCET, A. and PEREZ, P. (2003). Maintaining multimodality through mixture tracking. *2003 Ninth IEEE Internation Conference on Computer Vision* (vol. 2, pp. 1110-1116).

VIOLA, P. and JONES, M. (2001). Robust Real-time Object Detection. *Workshop on Statistical and Computational Theories of Vision.*

VIOLA, P., JONES, M. (2001) Robust Real-time Object Detection. *International Journal of Computer Vision*.

WANG, J., CHEN, X. and GAO, W. (2005). Online selecting discriminative tracking features using particle filter. *Computer Vision and Pattern Recognition (CVPR)* (vol. 2, pp. 1037-1042).

WANG, T.-S., SHUM, H.-Y., XU, Y.-Q., ZHENG, N.-N. (2001). Unsupervised analysis of human gestures. *Pacific Rim Conference on Multimedia, Springer-Verlag*, (pp. 174–181).

WANG, Y., JIANG, H., DREW, M., LI, Z.-N., MORI, G. (2006). Unsupervised discovery of action classes. *Conference on Computer Vision and Pattern Recognition* (vol. 2, pp. 1654–1661).

WEINLAND, D., RONFARD, R., BOYER E. (2011). A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, vol. 115(2), pp. 224-241.

WEINLAND, D., RONFARD, R., BOYER, E. (2006). Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, vol. 104(2), pp. 249–257.

WILKS, S.L., WOLF, W., LIANG, Y., KOBLA, V., BAIS, X., ZHANG, Y., CRNIC, L. S. Unified system and method for animal behavior characterization from top view using video analysis. (2004). U.S. Patent 7 643 655, Jan 5, 2004

WILLOW GARAGE. *Support Vector Machines*. Consulted on February 2013, taken from:http://opencv.willowgarage.com/documentation/cpp/support_vector_machines.html.

WILSON, H.R. and GIESE, S.C.(1977). Threshold visibility of frequencey gradient patterns. *Vision Research*, vol.17(10), pp. 1177-1190.

WUNSCH, P. and HIRZINGER, G. (1997), Real-time visual tracking of 3-D objects with dynamic handling of occlusions. *IEEE International Conference on Robotics and Automation* (pp. 2868–2873).

XING, H.J., HA, M.H., HU, B.G. and TIAN, D.Z. (2012). Linear Feature-weighted Support Vector Machine. *Fuzzy Information and Engineering*, 1(3), 289-305.

YANG, M.-H., AHUJA, N. (1999). Recognizing hand gesture using motion trajectories. *Conference on Computer Vision and Pattern Recognition* (vol. 1).

YAU, W., KUMAR, D., ARJUNAN, S. and KUMAR, S. (2006). Visual speech recognition using image moments and multiresolution wavelet. *Computer Graphics, Imaging and Visualization* (pp. 194-199).

YIN, Z. and COLLINS, R. (2006). Moving Object Localization in Thermal Imagery by Forward-backward MHI. *Conference on Computer Vision and Pattern Recognition Workshop* (pp. 271-291).

YIN, Z., PORIKLI, F. and COLLINS, R. (2008). Likelihood Map Fusion for Visual Object Tracking. *IEEE Workshop on Application of Computer Vision* (pp. 1-7).

YOGESH, R., VASWANI, N. and YEZZI, A. A. (2005). Particle filtering for geometric active contours with application to tracking moving and deforming objects. *Computer Vision and Pattern Recognition* (vol. 2, pp. 2-9).

YONG, R. and YUNQIANG, C. (2001). Better proposal distributions: object tracking using unscented particle filter. *Computer Vision and Pattern Recognition* (*CVPR)* (vol. 2, pp. 786-793).

ZHANG, C.X. and DUIN, R.P.W. (2011). An experimental study of one- and two-level classifier fusion for different sample size. *Pattern Recognition Letters*, 32, 1756-1767.