CHANGE-DETECTION WITH LIMITED SITUATIONAL AWARENESS

By

Christopher Costello

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Electrical Engineering

August, 2008

Nashville, Tennessee

Approved:

Professor Mitch Wilkes

Professor Richard Alan Peters III

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure Page

LIST OF TABLES

Table Page

ACKNOWLEDGMENTS

I would like to thank Dr. Mitch Wilkes for supporting, guiding, and encouraging me through this work. His ideas were invaluable in the completing this system, and his support and encouragement helped to motivate me through this work.

I would like to thank Dr. Alan Peters for being patient with me and helping me whenever I asked for it. Also, for the knowledge he passed to me in image processing and computer vision. This information will be very helpful in my future works.

I would also like to thank Mert Tugcu and Amy Wang for allowing me to use their work on this thesis. I would also like to thank them for the time they spent sharing the knowledge of these systems with me.

I give special thanks to Jonathan Hunter for helping me with this system when the others had graduated. His help as a resource with this system has been and continues to be priceless.

I thank Flo for always being there to make the good days better and the bad days good.

Finally, I would like to thank my friends and family. They have always supported me and encouraged me to do the best that I can. This support has been a great help in the completion of this work.

ABSTRACT

This work will focus on giving a perceptual system the ability to detect changes while maintaining its understanding of the environment. Most change detection systems can only perceive the change in the environment. They are not capable of processing the other objects in the environment. Nor are they capable of understanding what type of change they have just detected. This system aims to detect the difference between a novel object introduced to the environment and a known object moved within the environment while still segmenting the image.

The image segmentation will use very high dimensional feature vectors. These will be obtained from multiple training images, and each percept will be given a specific label. The feature vectors will then be converted into sparse vectors and arranged in an approximate nearest neighbor (NN) search tree. The new image's sparse vectors will scale the tree based on the Euclidian distances of the current sparse vector to the tree leaf nodes. The label from the leaf nodes will be selected as the representation of the percept in the new image. The novel objects will be detected based on a threshold distance from the leaf leave node. If this distance exceeds the threshold the object will be considered novel. The moved objects will be determined by a previously trained look up table (LUT). The LUT will hold a list of acceptable labels in for each pixel, and will be created from a series of training images.

The results from the experiments show that this system is capable of learning the objects in an environment and understanding how the environment changes.

CHAPTER I

INTRODUCTION

Change-Detection has much interest and multiple applications in a widespread group of disciplines. Some of those applications are video surveillance [13] [14] [24], medical applications [29] [30], and roadside observation [21] [22]. The video surveillance applications refer to security issues. These systems focus on finding changes in an environment and tracking the source of those changes. An example would be a person dropping a bag in an in a crowded environment [24]. The bag would be detected and associated to that individual. He would then be tracked around the environment. The medical applications are looking for differences over the course of time with respect to a patient. A specific example of this would be observing the evolution of lesions due to multiple sclerosis [29]. The third application includes automatically finding illegally parked cars [22] or finding vehicles that are stopped in traffic [21].

There are two limitations that are consistent throughout these applications. The first is that there is no situational awareness. This means that the system has no idea what it is actually looking at. All it can recognize is that there has been a change from one image to another. The objects throughout the rest of the image are not observed. For specific applications this is acceptable, however a general change-detection scheme should be capable of understanding it's surrounding and also understanding the type of change.

This brings up the second limitation in of these systems. There is no means of understanding the type of changes. Because all of the previous examples are very task specific, assumptions are made as to the type of change. This means that they may not understand if a novel object has been introduced or an object in the background has simply moved.

The goal of this work is to build upon the perceptual system designed by Tugcu [1] and Wang [2] and add the appropriate change-detection features, removing the stated limitations. The final goal is a system capable of understanding its environment, and able to provide useful feedback to the users. An example would be, but is not limited to, airport security. All of the features in the quiet environment should be understood, and their proper locations known (e.g., plants, paintings, and benches). A quiet environment constitutes a situation with little to no movement. This would be when the airport is closed or early mornings when only a few people are present. When the environment changes, the system should understand how it has changed. It should be able to determine and understand the difference between a package being left and a plant simply being moved.

Furthermore, the system should be capable of determining transient high activity times of the day (e.g., heavy crowds walking through the terminal) as well as quiet times. When the area is going through rapid changes the information should be considered as largely unimportant because of the large amount of change. Even humans have a lot of trouble with this. When the crowd dissipates the scene should be processed again and any new objects found or previous ones removed should be detected.

Another feature is to incorporate a priority to the objects being detected. An example would be the wall of the terminal. Because it does not move, it does not change and thus the only information it provides is a boundary to the room. The system should be able to learn and understand these basic elements of the environment allowing it to process them faster. The opposite ides to this would be short term temporal changes, such as people walking by, or objects moved.

The short term changes should be the main focus and have the highest priority. An example of this would be a briefcase left behind. In an airport, security would want to be made aware of this immediately.

The short term detection should be able to determine the importance of the change. If an unidentifiable object has been left then the system should notify the users with a message marked with high importance. However if a plant or chair has been moved or removed, there is no need to sound alarms. The system should simply notify the users of these detected changes with a message marked with lower importance.

The final characteristic should be the ability to incorporate the changes over time. If a plant is moved and it is determined not to be significant, it should be incorporated into the background, i.e., associated with the background.

This work will focus on adding the novel object detection feature and moved object feature to the system built in [1], and how to discriminate between the two types of changes.

CHAPTER II

RELATED WORK

True autonomous robotic vision has not yet been achieved. Many of the methods previously created are very specific and non-robust. Often it is very difficult to teach those systems new percepts. Because of this limitation, they will fail in new environments. The ultimate goal of this system is to be generally applicable in any environment, and to have the ability to discern the difference between novel object changes and moved object changes in the environment. The perception system created by Tugcu [1] and Wang [2] has been chosen as the starting point for this project.

Origin of this Perceptual System

Tugcu [1] created a very robust and capable visual system and combined it with the working memory toolkit [5] in order to perceive the environment and decide the proper action. Because perception is the priority of this work only that aspect of the system will be discussed.

The perception system begins with training. The user must select each percept and give it a label. The pixel values are then used to create a high dimension feature vector of 10,001 bins. The first 10,000 bins are a histogram representation of the hue, saturation, and values in a small region of the image. The last bin is a texture feature that provides contrast and is calculated from a filtering technique based on Laplacian operators. Because of the small size of the image region, the majority of the elements in

the feature vector are zero's. This allows a sparse vector representation to be used. From the sparse vector database a 3-way approximate nearest neighbor (NN) search tree is created. The nodes of the NN search tree are clustered based on the Euclidian distances of the sparse vectors from each other.

Once a NN search tree is created, a new image is captured. This image is then converted into feature vectors in the same way as the training data. The feature vector then chooses the child nodes of the tree based on the Euclidian distance of the feature vector to the centroid of the node. Once a leaf node is reached, if the node is made up of feature vectors of the same label, that label will be used to classify that area of the image. If the leaf node is made up of more then one label, an exact NN comparison is done with each feature vector in the node. The label of the feature vector that is closest to the current feature vector is then selected to represent that part of the image. This implementation will be described in more detail in Chapter 3. Figure 1 shows an example of this segmentation.

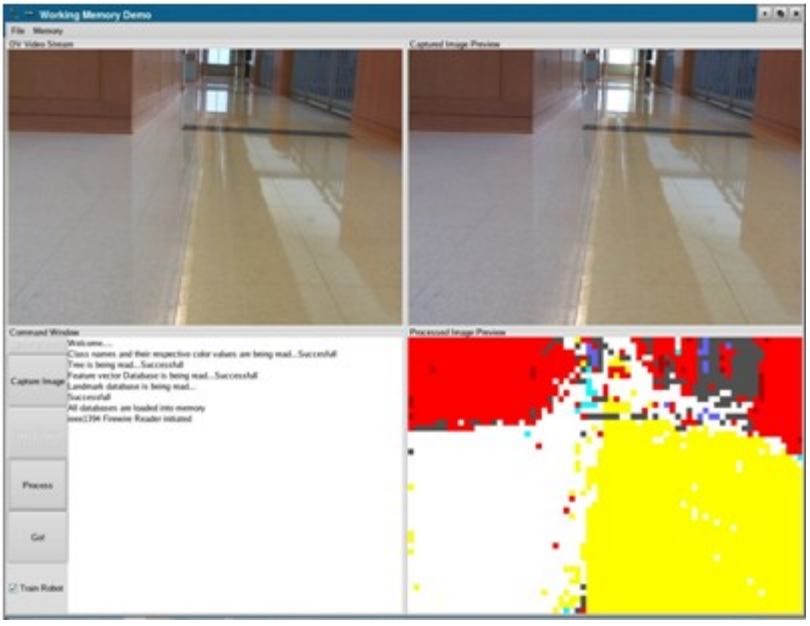

Figure 1[1] Example of a segmented hallway image

The features of the hallway are segmented and represented in the bottom right. The end of the hallway has a bright reflection in it causing the noise. Because this system is based on color that much of a change is expected to impact the results. What makes this system particularly desirable is its potential for expandability.

Currently, one limitation on this system is that when a new object is trained, it is not efficiently added to search tree. The new information will be added into a leaf node. This creates the possibility of very large leaf nodes that can greatly increase search times. The first change implemented will be to adjust the tree so that it can expand properly in real time. This will allow for immediate results after training without having reduced speeds or having to completely rebuild the tree.

Another constraint is that there is no change detection. If a new object were presented in the environment the percept the object is closest to will represent it in the segmented image. This will require two new features. The first feature is the ability to detect novel objects; the second feature is the ability to detect subtle changes in the present objects. The novel object detection has been implemented on a similar system created by Wang [2].

Wang [2] created a similar tree structure. The differences are that in [2] the texture features are calculated differently, and the percepts are decided autonomously from a training set of images. The texture features are calculated based on Gabor filter. The Gabor filter used in [4] is based on a complex exponentials with a Gaussian

envelope. This calculation provides 40 texture feature bins creating a total feature vector size of 10,040.

The training method in [2] is also different. A series of images are presented to the system and the clustering of percepts and decision tree creation are all done without user intervention. Figure 2 shows the performance of this system.

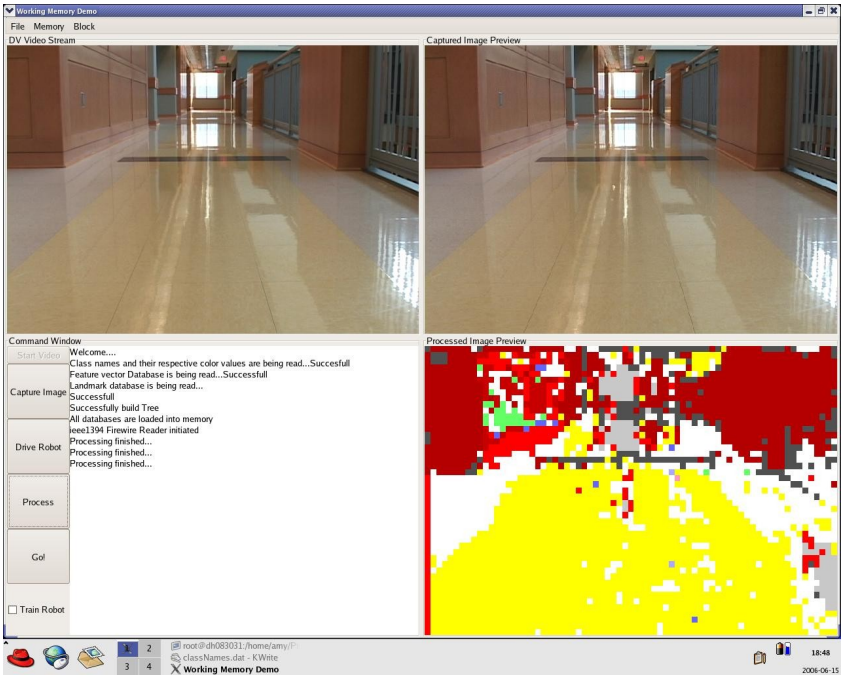

Figure 2 Example of the performance of [2]

This shows that this system is comparable to the performance of [1]. The main percepts in the hallway have been segmented out and can be discerned in the image.

The application of this system that is of most interest is novel object detection. This was accomplished by using distances from both an approximate nearest neighbor (NN) search tree and a minimum spanning tree (MST) [2]. The first process finds percepts in the NN search tree. The feature vectors are then processed through the MST

and a “low pass” filter is used. If the distances from the nodes in the MST are below a threshold the percept from the NN tree will represent that pixel. If the distance is greater than the threshold the feature vector is kept as a possible novel object. Once possible novel feature vectors are found, erosion with an 8-connected struture element is performed twice. Figure 3 shows the novel object detection’s performance.

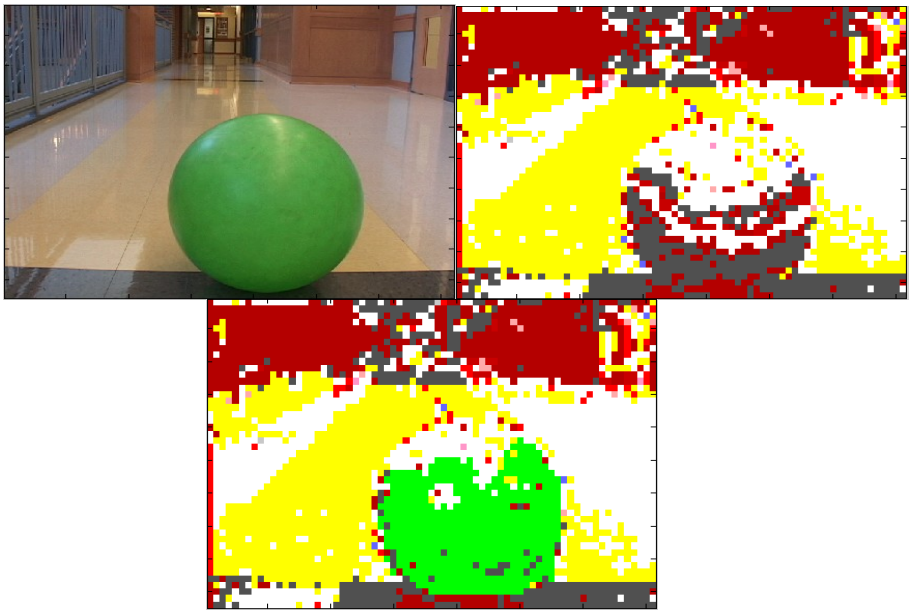

Figure 3[2] (top left) the original image, (top right) processed image with NN tree, (bottom) processed images after learning with MST.

This is an example of the system detecting a novel object and autonomously adding that object to the list of percepts. This ability is nice but not desired in the system being created. The novel object detection in this work will find the novel objects and alert the user to their presence. It will operate under similar a threshold principle, however it will not automatically accept and learn the novel object. Another difference is that [2] assumes that only one novel

object has been introduced to the scene. This work will allow for multiple novel objects to be detected simultaneously.

Related work in Change-Detection

The first and most obvious application of change-detection is video surveillance. The first application which has been done multiple times is detecting an abandoned object [14] [24]. In all of these applications the object is attached to the owner before it is dropped. The owner and object are then tracked by the system.

The system in [24] has broken down the process of detecting an unattended package into four steps. The first step is moving object detection, tracking and classification. The moving objects are detected using a static change detection method (SCD) from [25]. This method uses a threshold to determine if an area based on a sliding window is moving. If the area is moving it is represented as a blob on a binary image.

Eigen-features are then extracted for object classification. The Eigen-features are used because they provide more shape-based information and are a more compact representation. These vectors are used to represent the original image in a lower dimensional space.

A support vector machine (SVM) is trained with five classes of human motion: walking human with a bag, walking human without a bag, bending human, human separating from bag, and unknown class. This will be applied to the real-time tracked objects detected for classification.

It is also desired to track multiple objects with and keep track of them. The method created by [26], was extended using Kalman filters and shape and color matching. The Kalman filter predicts the position, dimension, and changes of each moving object from noise of occlusion. In the case of occlusion, the shape and color of previous images is used for matching. The SVM is applied to these results to generate a symbol tracking the human motions.

The second step in this process is assigning ownership to the packages detected. This is done using hidden Markov models (HMM). The Baum-Welch algorithm [27] is used to optimize the parameters for training the HMM. One HMM is created for each of the activities of interest. Then when the sequences are observed the HMM that it matches most closely is selected as the human activity that lead to the unattended package.

The third process is determining the object location for a distance relationship to the human. The Ray Tracing [28] technique originally devised to project a three-dimensional world into a two-dimensional representation was inverted for this process. The image is already in a two-dimensional world so it needed to be projected into a three-dimensional world in order to obtain the distance relations. Once the distance from the human owner to the package was determined the final step was performed.

The final step is determining that a package is unattended. This is performed by determining package ownership, the spatial distance, and the time distance. The system tracks every moving object. When a package is determined stationary all humans nearby are considered possible owners. The symbol

sequences previously processed are run though the HMMs to recognize which of the humans dropped the bad. Once an owner has been established the distance is checked for a 10m threshold. If the owner does not return within the threshold distance in 15s an alarm will sound. The 10m and 15s were predefined values given to the system.

This system was tested and worked well with four humans in the image and one bag. There was no report on how well the system would perform with multiple bags. Also the system works well in light traffic areas, however if heavy traffic were to travel through the visual field this system would likely fail. The use of immediate difference images would make it very difficult to determine if one individual in a crowd left a bag even after the crowd dissipated. Our proposed system will be able to handle this event.

The next system to review was created for determining illegally parked cars in the road [22]. This system also uses four processes: one-dimensional projection, segmentation, tracking, and reconstruction. The one-dimensional projection is used to convert the no parking zones into one-dimensional vectors. These one-dimensional vectors are a means of representing the curved no parking areas in a straight line. This will allow simpler assessment of whether a car is parked there or moving through.

The segmentation is performed by combining background subtraction and differencing frames in a temporal sequence. To initialize the background it is assumed there are no cars parked. Then the sequence of images is processed performing morphological operations to remove noise.

The third step is tracking the objects. Tracking is performed by matching feature distances. The features are length, location, RGB color, and velocity. These features are used and projected onto another image to represent the tracked object.

The final process is the reconstruction of the image from the one-dimensional representation. This is done to verify that the cars are parked in the no parking zone.

This system performed very well in detecting the illegally parked cars. However, once again this system is dedicated to only one process. Because the no parking zones of the image need to first be mapped, this system is severely limited in practice. Many of the systems discovered during the survey were similarly limited in generality. They use similar background subtraction with some temporal function to determine a change and then perform various mean of tracking. These systems work very well in a single environment, but they cannot easily be applied anywhere else.

CHAPTER III

IMPLEMENTATION

Overview

The proposed system will have numerous parts. Various parts of the initial system are also covered in Tugcu's [1] and Wang's [2] works. It begins with the very high dimensional feature space used to discern the different objects. The feature vectors are then classified using a decision tree, which greatly reduces the search times for classification. The system then determines if the objects segregated are novel objects and if they are expected to be in their locations. The data flow for this system is shown in Figure 4.

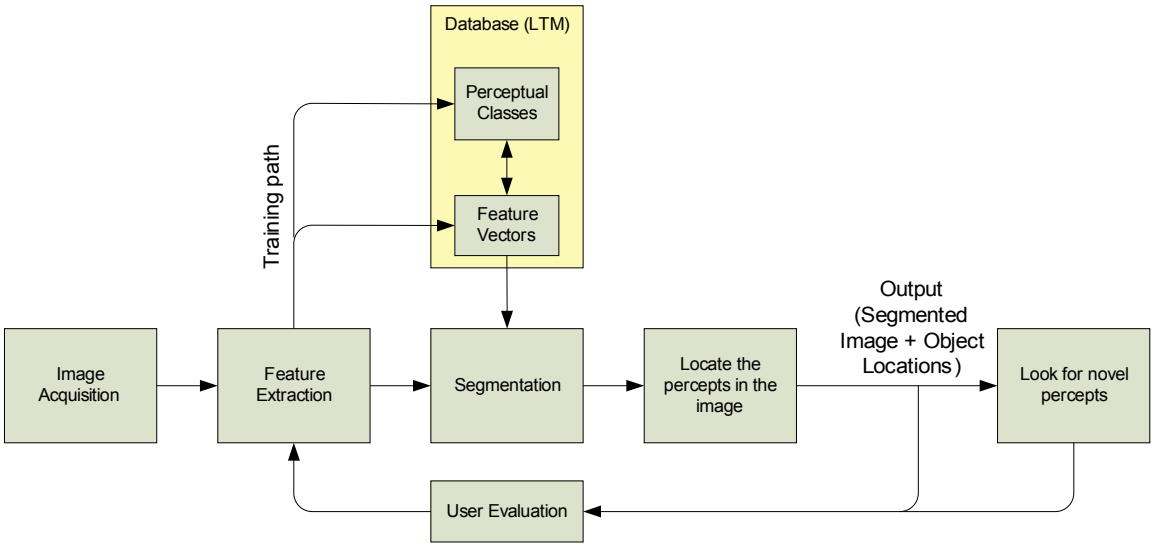

Figure 4[1] Flowchart of Perceptual System

This flow chart shows that initially the user will teach the system the labels and percepts. The system will then identify the important information to learn the percepts. This process is repeated until there is sufficient information to classify each of the percepts. The robot will then capture a new image and segment it resulting in an image with its regions classified. If the robot has determined there to be a novel object or an object out of place the image will contain that information. The human user then determines if the percepts have been identified, and if the novel objects or moved objects need to be learned by the robot. If it is determined that the robot should learn more information, it can be trained additionally at that time.

Very High Dimensional Feature Space

The system is chosen to have a very high dimensional feature space in order to allow for a high capacity to learn. In this case the feature vector will represent the color distribution in a small region. This allows for subtle discrimination amongst colors to deal with shadows, patterns, and lighting variations.

A very high dimensional feature space does not come without its drawbacks. The first problem is the size of the database required. The rule proposed in [32] suggests that five times the size of the feature vector is a sufficient amount of data. This is not a major problem for this system because it is easy to obtain the required amount of training data through the images.

Another problem with high dimensional feature spaces is that parametric classifiers become less useful as the number of dimensions rises [1]. These parametric techniques include calculating Eigen-values, Eigen-vectors, covariance matrices,

acquiring covariance matrices, or forming large data matrices and calculating singular value decompositions [1]. The calculations to perform any of these techniques with very high dimensional feature spaces are very difficult and very expensive. This leads to the use of the nearest neighbor (NN) classification technique, which only requires the distance calculations in order to be effective.

The drawback to using NN classification is that if the data is simply collected and exhaustively searched it will take too long and be too computationally expensive. In order to speed the processing up an approximate NN search tree was applied. This provided robust performance at a greatly reduced processing speed. Another benefit of the search tree is that the system can be trained in real time much easier then for parametric approaches.

In order to deal with the computational expense of the large feature vector, a sparse vector representation is used. For this system color histograms are used which, when extracted from small regions of the image, can be very well modeled with sparse vectors. This both simplifies the calculations to be performed and reduces the space required for storage.

As mentioned, the high dimensional feature space is a representation of the color in the image. The system will first grab the image and obtain the red, green, and blue (RGB) pixel values. These values are then converted into Hue, Saturation, and Value (HSV). The HSV coordinate system is shown in Figure 5.

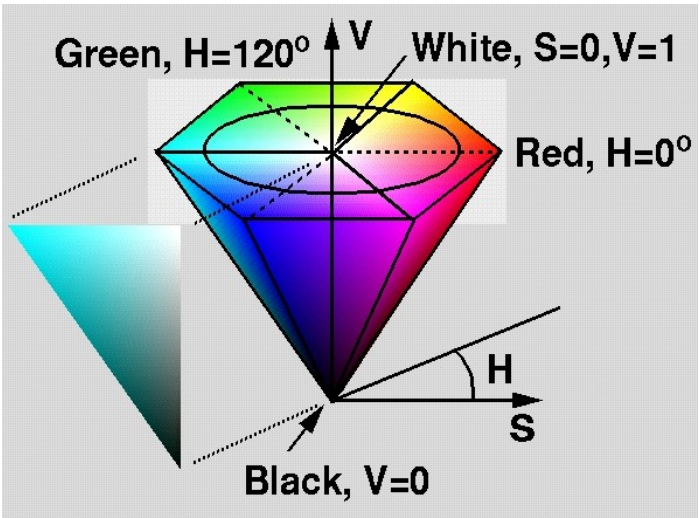

Figure 5 [1]. HSV color domain representation.

Hue defines the color and is represented from 0º to 360º. The saturation is the purity and is represented from 0 to 1. The value is the intensity of the color and is also represented from 0 to 1.

The conversion form RGB to HSV is done as follows.

$$MAX = \max(R, G, B)$$
$$MIN = \min(R, G, B)$$
$$V = MAX$$

$$if\ (MAX = 0),\ then\ \ V = S = 0\ \ and\ \ H\ is\ undefined$$
$$S = \frac{MAX - MIN}{MAX}$$
$$if\ (R = MAX),\ then\ \ H = \left(0 + \frac{G - B}{MAX - MIN}\right) \times 60$$
$$if\ (G = MAX),\ then\ \ H = \left(2 + \frac{B - R}{MAX - MIN}\right) \times 60$$
$$if\ (B = MAX),\ then\ \ H = \left(4 + \frac{R - G}{MAX - MIN}\right) \times 60$$
$$if\ (H < 0),\ \ \ \ then\ \ H = H + 360$$

Once the HSV values are computed a probability density function of the distribution is made. The hue of the histogram is broken into 100 bins while the saturation and value are each broken down into 10 bins. By combining these results, the total number of color features is 10,000. The HSV values of small regions are then taken and broken down in to 15x15 regions to form each feature vector. This leaves a large part of the feature vector filled with zeros. Because of this, a sparse vector is an efficient representation of the vector.

The sparse vector has two parts. The first is a value vector. The value vector provides the number of pixels read that contained that value. The second part is the index vector. This vector contains the indices of the locations of the values in the original feature vector. These vectors combined represent all of the relevant data in the high dimensional feature space. This representation reduces the computational cost of calculating the distance between the feature vectors.

The reason for this is because the norm of the vector can be calculated with only non-zero elements. Also, the inner products of the two vectors only need the values where both vectors have non-zero elements. Therefore as long as the number of non-zero elements is held close to constant, the calculations will not grow. The system restricts the area searched to an N x N region. Thus, the maximum number of non-zero elements is going to be $2N^2 + 1$. The one comes from the texture feature included.

The texture feature is included to provide a measure of "roughness" vs. "smoothness" in the image. To calculate this value a spatial filtering technique, based on a Laplacian operator, is used [1]. The Laplacian operator will highlight the edges in the image.

Search Tree

In the previous systems [1] [2], a large training database had to be created and the search tree was then created from this database. When new feature vectors were added to the tree, the leaf node that they were closest to was found and then the feature vectors were added to that node. The problem with this is that as more feature vectors are added the larger the population in the leaf node would become. As the node became larger the speed benefits of the approximate NN search tree were mitigated. The way around this was to create a whole new tree from the entire new database. For the proposed system it was desired to have the robot learn from a top down approach. This means that the database and tree expand at the same time. The first vectors trained became the top node of the tree. As new feature vectors and percepts were trained, they were immediately added to the tree. The structural means by which the tree is broken down is still the same as in [1] [2]. The difference is that the current tree will provide immediate feedback to the new training.

The tree is structured into a 3-way approximate NN search tree. It will start at the root node and when the new training vectors are added, the system will cluster the vectors accordingly and create three child nodes around the centroid of the three clusters. The general form of the tree is shown in Figure (6) [2].

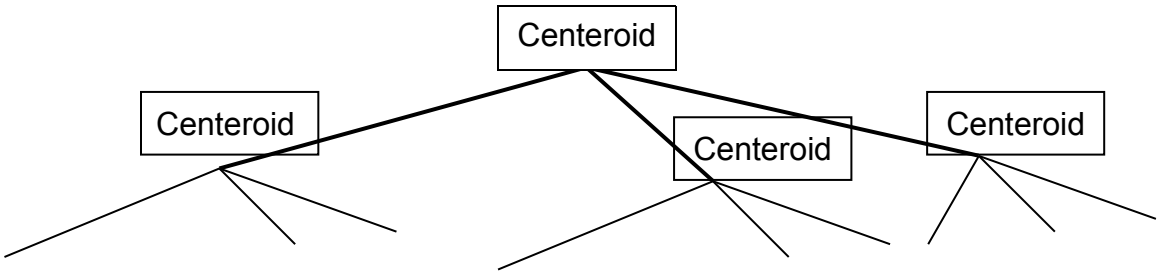

Figure 6 [2] Illustration of database tree structure

The clusters are decided based on proximity calculations. According to [31], the best approximation for invariant relations between data and distances for HSV is the Euclidian distance. The K-dimensional space Euclidian distance is calculated by:

$$d_{ij} = \left(\sum_{k=1}^{K} \left| x_{ik} - x_{jk} \right|^2 \right)^{1/2} \quad [1]$$

As mentioned, the number of dimensions K will be 10001 in the system.

When an image is being segmented it will calculate the distance from the current image feature vector to the center of each of the child node clusters. It will then follow the path of the node closest to the feature vector, down the tree until it reaches a pure node or it reaches a node with no children. A pure node is a node that has only one label associated to it. If this node is selected then no further calculations are necessary. The label corresponding to this node is taken. If the process reaches a leaf node that is impure, it will perform a NN search in this node. It is because of this NN search that simply placing new feature vectors in nodes without expansion of the tree can increase search time.

A maximum likelihood estimator (MLE) decision process has also been recently implemented in the tree structure. An MLE is a probabilistic determination of which class the current feature vector should be placed in when at impure nodes. The class that has the highest number of feature vectors in the impure node will be selected as the current feature vector. The progression through the tree remains the same, however when an impure node is selected this will save calculation time over the NN search. This also opens up the possibility for the system to provide more useful feedback regarding the reasons for its selections. The further investigation of this system will be left for future work.

Novel Object Detection

The novelty detection in this system is based on a threshold applied to the K-dimensional Euclidian distance with the current feature vector and the centroid of the node that it is determined to be closest to. The threshold was determined by taking 10 sample images and calculating the mean and standard deviations of the distance from the pixels to their closest nodes in the tree. Then two times the standard deviation was added to the mean to get threshold value. This threshold was then fine tuned for the system through trial and error experimentation.

The problem with this novelty detection scheme is the amount of noise produced in the image. If one pixel is determined to be noise, it is undesirable for the system to continually respond. It should only respond if there is a new object present. The technique used to rid the image of noise was a size constraint. If a pixel is determined to be outside of the threshold all of its neighbors are checked. If a total of seven connected pixels are determined to be outside of the set threshold then a novel object has been detected and the user is notified.

Moved-Object Detection

The primary difference between this feature and novel object detection is that this should recognize if an object the system already knows has been moved in the background. This should operate similarly to a surveillance system. If an object in the image that the system knows has been moved, the system should be able to understand that it has moved. It should then notify the user of the movement.

The purpose of this ability is to provide context to the environment. When humans enter a room with a desk and a chair it does not matter if the chair is at the desk or beside the desk. However if the chair has been moved to the wall on the other side of the room, this would be considered odd. Also if the chair were to be completely removed from the room, humans would notice.

This function is performed using a look up table (LUT). For each pixel a list of acceptable labels are placed in the LUT. This is done by using a series of training images with objects in acceptable locations. Multiple images are used for the training to reduce the effect of noise and allow objects to be slightly shifted. Then when the new image is segmented, the pixels are compared to the acceptable values. If a pixel has an unacceptable value its surrounding pixels will also be checked to determine if they are new values. If seven or more of them are new values it is determined that an object is moved. Because of the presence of transient noise for this system, it must process the next image and if the same pixels are determined to have unacceptable values then an object is determined as moved.

CHAPTER IV

EXPERIMENTS AND RESULTS

The system described has been implemented on the stationary humanoid robot (ISAC) at Vanderbilt University. A camera has been mounted in front of ISAC to see the room directly in front of it. A series of 10 images were taken of the environment and the percepts to train on were decided. The 10 images were taken over multiple days at different times. This was an attempt to mitigate the effects of shadows changing throughout the day. Figure 6 shows the environment the system will be learning.

Figure 7. Image directly in front of ISAC taken at 3pm.

The features chosen are the wall, floor, power strip, white erase boards, trash can, printer and chair. The other objects, e.g., the table, are not selected because they are not large enough to train on. For the training images all of the objects were reliably

segmented. However when the system was run outside of the training images, it was found that the white nature of the wall, white-boards, and printer and the black nature of the chair and trash can were indistinguishable throughout the day. The accuracy of the system was dependant lighting variations on each of these objects. The decision was made to combine the white objects into one percept and the black objects into another percept. This left four identifiable percepts in the environment: the white object, black object, power strip and the floor. Table 1 lists the colors that will represent the features in the segmented image.

Table 1: List of percepts and representative colors

Object(s)	Color Representation
Wall, White Erase Boards, Printer	Gray
Floor	Black
Trash Cans, Chair	Green
Power Strip	Orange
Novel Objects	White
Moved Objects	Red

Experiment 1

The first experiment will demonstrate the novel object detection. This will show how the system responds when various objects are introduced into the scene. The first object that will be placed in the scene is a red tool chest. The image and results are shown in Figure 7.

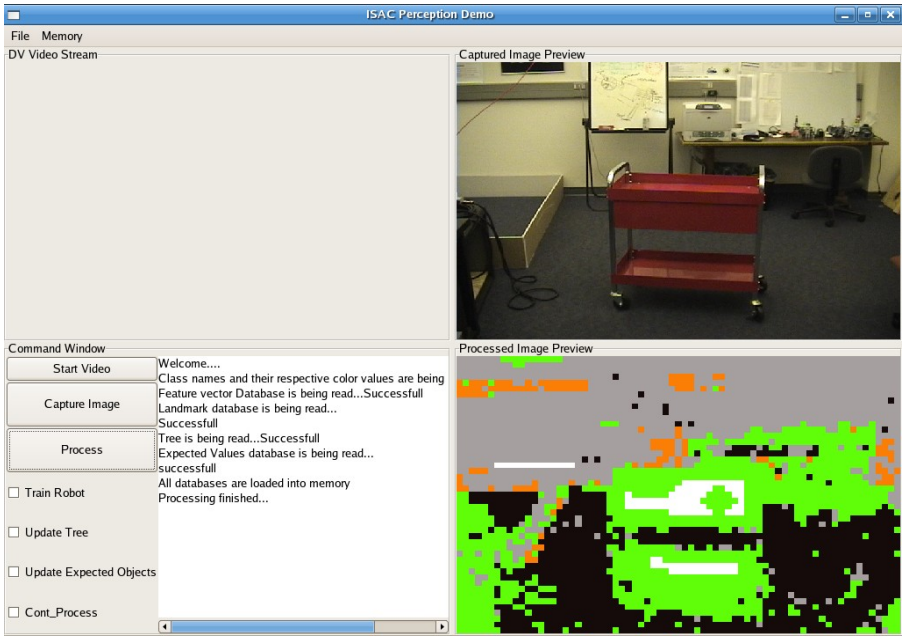

Figure 8 Processed image with novel object introduced.

The processed area shown in white is the area detected as novel. It is shown at the center of mass of the object. The surrounding is green because of the erosion. During the process the eroded areas are classified as the object they are closest to. In this case and many others it is the trashcan/chair percepts. Another thing to note is the novel object detected at the bottom of the white wall. This phenomenon will be discussed later. Another example of novel object detection is shown in Figure 8.

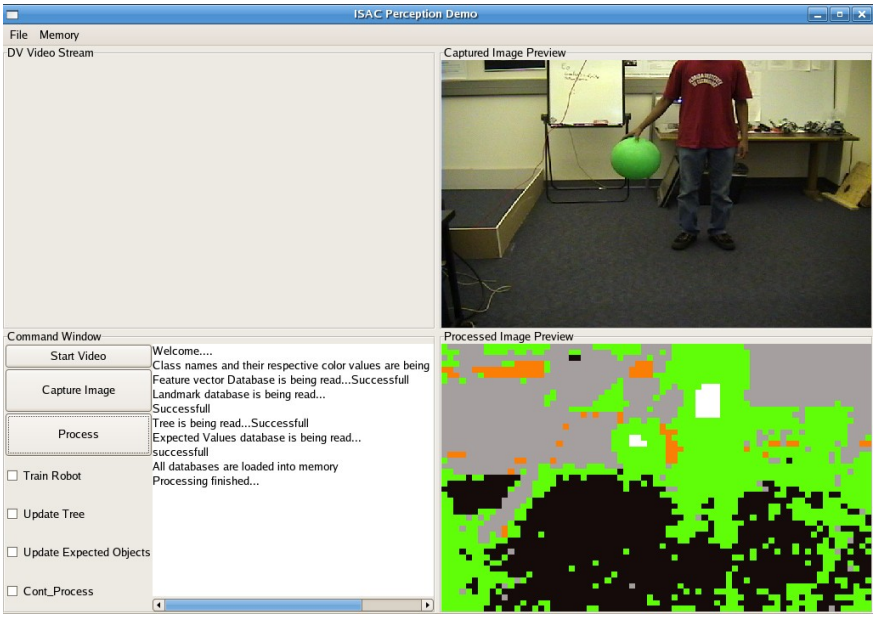

Figure 9 Multiple novel objects

In this example both the green ball and red shirt of the individual holding the ball are detected. The rest of the body has been highlighted as closest to trashcan/chair again. A series of objects have been tried. Those that had color not appearing in the environment normally were recognized while those that had colors similar to the normal environment were detected as moved objects. An example is the blue jeans worn in Figure 8. They are not perceived to be novel. However if the individual remains in that location long enough to be processed again, they would be reported as a moved object. The second process is necessary because moved objects are only reported after being detected in two consecutive images.

An interesting event repeatedly happened during this process. That event was that no matter how much the white areas were trained they would randomly be detected as novel objects. The reason for this was determined to be the lack of saturation throughout the image. As mentioned in Chapter 3, the hue was given the most bins with 100.

Therefore, it has the highest impact on the distance calculations. The problem is that when there is no saturation and high value the image will still appear white regardless of the hue. This means that the distances may be relatively far away even if they appear to be the same colors. A possible solution to this would be to increase the number of saturation bins in the histogram. This will be left for future work.

Experiment 2

The second experiment will demonstrate the surveillance capabilities of the system. The first test will be moving the printer. Figure 9 shows the results.

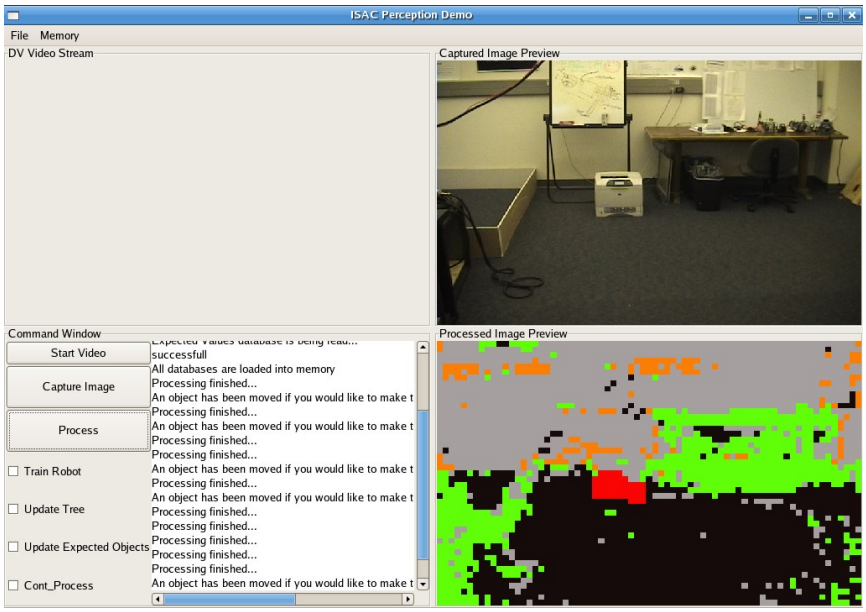

Figure 10 The printer moved to the floor.

The printer has been highlighted in red instead of the typical gray that defines it. A message has also been printed in the text area to let the user know of the moved object.

The next test, Figure 10, shows what happens when the trash cans have been moved. To demonstrate that the system can handle moved objects and novel objects at the same time, a pink object has been added.

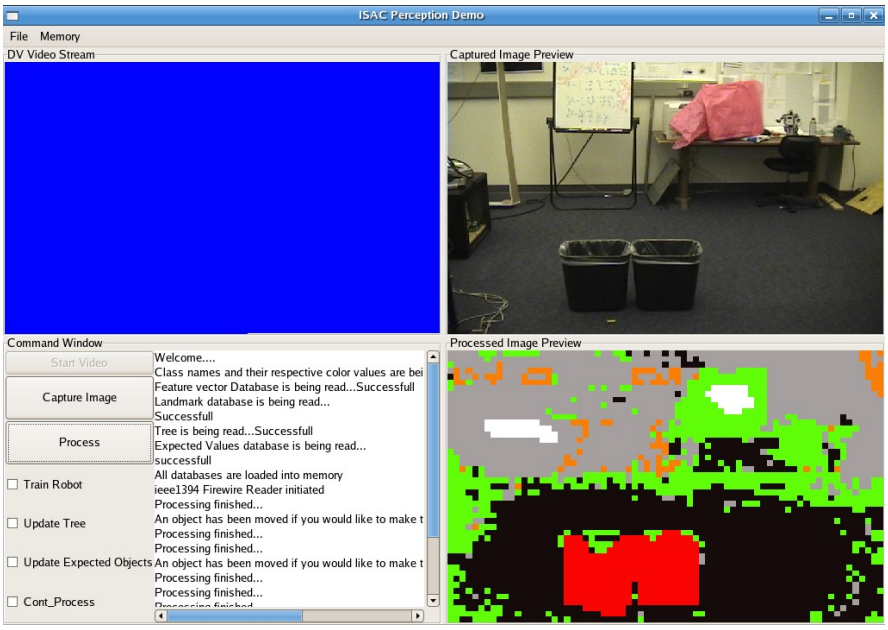

Figure 11 The trash cans have been moved and a pink box has been added.

This image shows the system handling both events simultaneously. Also the issue previously mentioned of a false reaction to the white wall can be seen more clearly here.

The third possibility is the complete removal of an object from the image. This is shown in Figure 11.

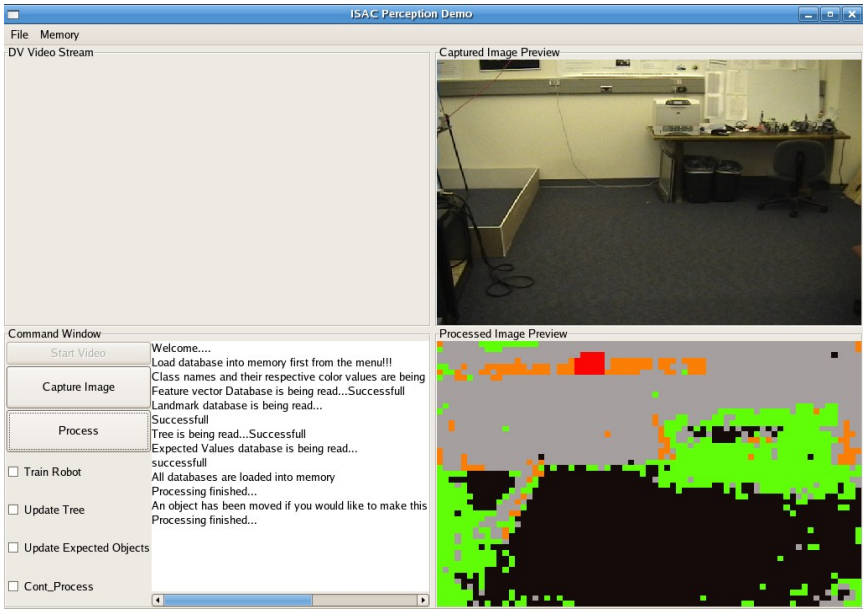

Figure 12 White erase board removed.

This example shows the moved object noticed in the power strip. Because the wall and white board were combined into one object, it is not expected to be detected anywhere else.

The final example is one in which the system will fail. Because of the large amount of combining percepts in this environment, it became easier to trick the system. Figure 12 shows this failure.

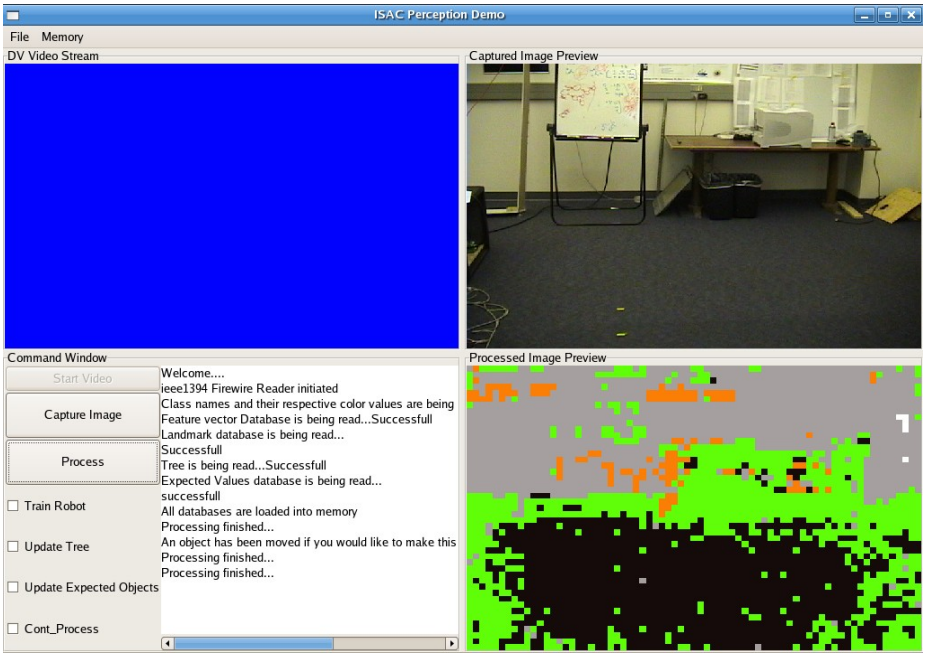

Figure 13 Printer moved in the image and the chair removed from image

This shows that when the printer is moved against another white background, the movement goes undetected. Also, the extremely dark shadows under the table allow the absence of the chair to go undetected. This failure shows that in a relatively colorless image, change is difficult to detect.

These are four examples of cases where objects were moved in the image. As more tests were performed, the objects that were novel objects and not detected as novel were still noticed by the surveillance system. The system believed that they were in the trashcan/chair category but realized that they were in a location not expected.

Experiment 3

The purpose of creating the MLE decision tree is to speed the system up and allow the tree representation to be smaller. To run a single iteration of the approximate

NN search tree takes 12 seconds. Also, the current tree structure exceeds 100mb of space. It is believed that the MLE representation can greatly reduce both of these.

Three different tests are compared with the MLE tree. The first is how long and accurately the system performs with the full MLE tree. The second test performed is with only 45 levels of the tree. The final tests performed, goes down 30 levels of the tree. The results of the first test, i.e., the full tree, are shown in figure 13.

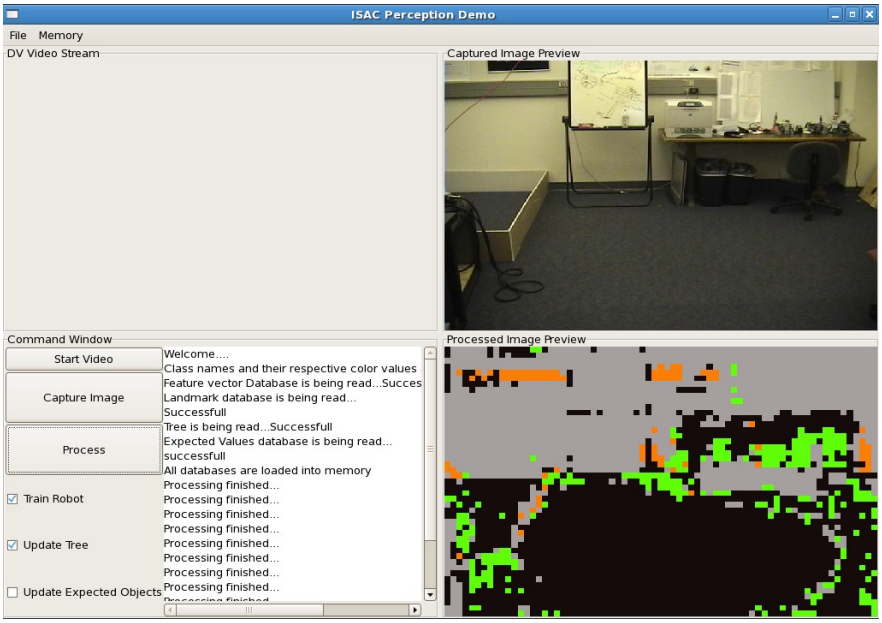

Figure 14 Processed image with full MLE tree.

The system took 5 seconds to process this image. This result shows that the process took less then half the time of the NN search, but the loss of quality is evident in the trash can and chair. The NN search was able to more reliably detect the percepts under the table.

The results of the tree going only down to the 45th level are shown in Figure 14.

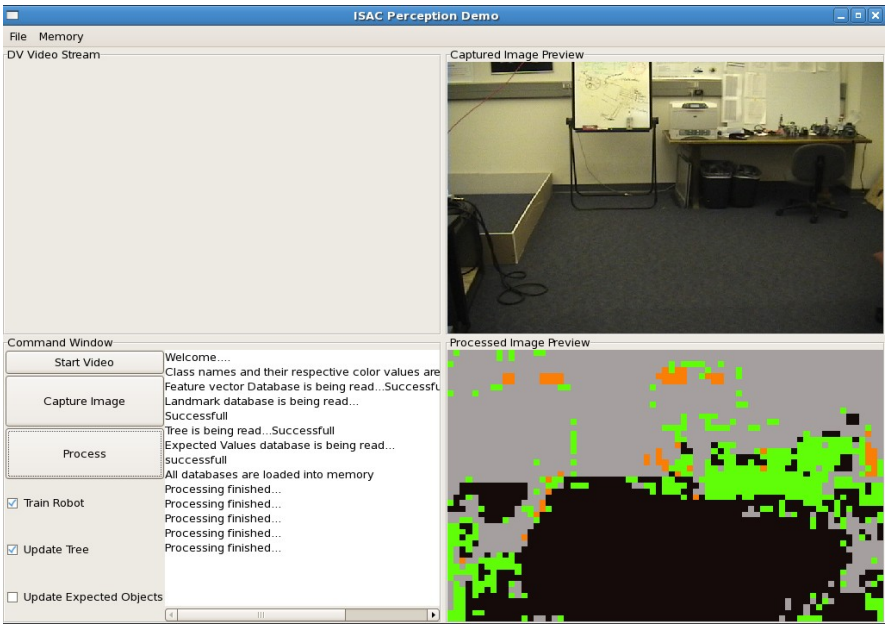

Figure 15 Processed image with 45 levels of the MLE tree.

These results were obtained in 4 seconds. They also provided results consistently more accurate than the full MLE tree. However compared to the NN search the power strip is nearly nonexistent.

The third experiment with an MLE tree is shown in Figure 15.

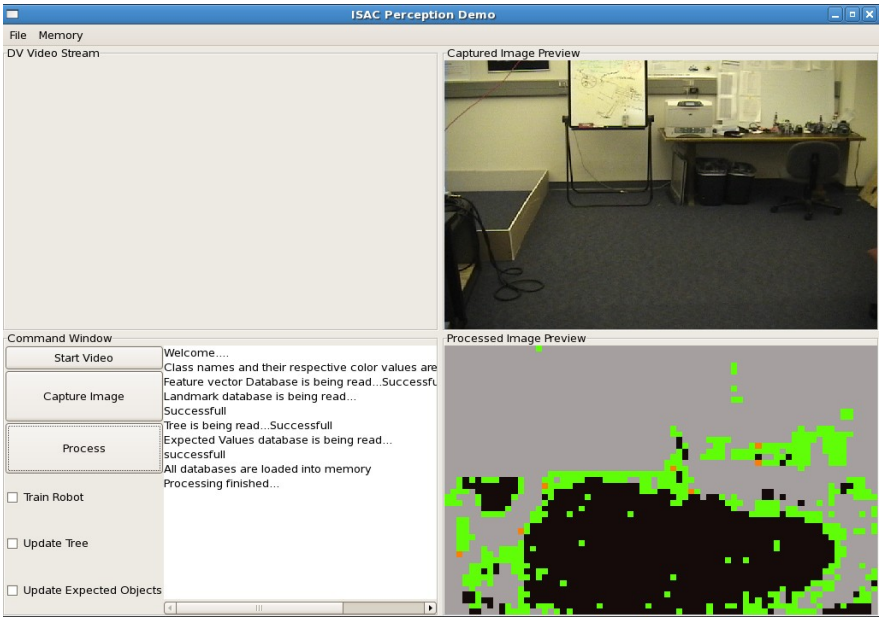

Figure 16 Processed image with 30 levels of the MLE tree.

This result was obtained in 3 seconds and was very noticeably different from the full tree result.

The various results demonstrate the speed up of the system along with the varying accuracy. Using the tree to the 45th level was repeatedly the most accurate representation while providing the 2nd fastest results. The purpose of demonstrating the MLE tree at 30 levels is to give an idea of the usefulness of this tree in a very simple environment. The wall and floor are represented reasonably well in only 3 seconds. This provides reasonable results if speed is more important than accuracy.

CHAPTER V

CONCLUSION

This work has shown the flexibility of using a very high dimensional feature space representation for change-detection in vision. The distinction between novel objects and moved objects was shown. These changes were also detected within the context of the image. The knowledge of the background was not sacrificed in order to pick up the changes.

The novel object detection was performed using a threshold based on the distances of the feature vectors to the tree nodes. A size constraint was then applied to remove noise. This worked well except for the areas with very low saturation. As was mentioned because of the lack of saturation the hue, which carries the highest weight in distance calculations, became meaningless.

The moved object detection was performed via comparison to a LUT. The LUT gave a sense of objects that belonged in an area. This allows chairs to be moved under a table without issue, but would detect a chair in the middle of the room where one is not expected. One possible next step of this detection scheme is to have the system learn to understand objects that cannot move so that they may be considered less important and have less processing time devoted to them.

A number of improvements can still be made to this system. The first improvement would be autonomous clustering. The ability to place the system in any environment and have it reliably figure out what is important would be very useful. This

process was started in [2]. A random feature vector was selected and declared to be the centroid and from there a minimum spanning tree was generated. The results from the experiments indicate that this could be a good starting point for this addition.

Another addition to this work would be the concept of time. There are two notions of time that relate to this system. The first learns when the transient times of day are, i.e., the times of day when the environment undergoes the most rapid changes. The system is capable of determining when a transient time is by the number of changes that are constantly happening. During this time the quickly changing areas of the room do not need to be processed. When the area is quiet again, the room should be processed for all of the changes. The second idea of time is when an object has been left in a single location for a long time. An example would be a picture on the wall. The picture does not tell you anything new about the environment and it is therefore undesirable to spend a lot of time processing it. The significance of the picture should drop and it should be integrated into the background.

The next means of improving this system is a coarse vision combined with a focused vision system. The coarsely segmented image is a result of the 15x15 windows used in creating the feature vectors. The focused segmentation would use a smaller window size and provide more details in a select area of the image. By including these functions, the processing of the irrelevant background can be sped up while the focused section can provide more detail on the areas of interest in the image.

Another addition will be to fix the issues with the low saturation areas. One idea as a solution is to increase the weight of the saturations. It currently only has 10 bins associated to it. If this were increased to 100 bins the saturations could be analyzed much

closer. This solution will also not affect the speed of the system. The processing speed is based on the 15x15 moving window not the total size of the feature vector.

The final improvements to be worked on are the MLE trees. The purpose of creating them is speed and size. The next step with them is to reduce the trees while retaining the appropriate information. This will determine how they can be applied. Ideally the tree will be reduced enough that the tree will be able to fit on the graphics processing units (GPUs) on video cards. This will greatly increase processing speed, making this system closer to real time video processing.

REFERENCES

[1] Tugcu, M., “A computational neuroscience model with application to robot perceptual learning”, Ph.D. Dissertation, Vanderbilt University, August 2007.

[2] Wang, X., “A Vision-Based Perceptual Learning System for Autonomous Mobile Robot”, Ph.D. Dissertation, Vanderbilt University, August 2007.

[3] Tugcu, M., Wang, X., Hunter, J.E., Phillips, J., Noelle, D., and Wilkes, D. M. "A computational Neuroscience model of working memory with application to robot perceptual learning", *Third IASTED International Conference on Computational Intelligence* (CI), Banff, Alberta, Canada, July 2-4 2007.

[4] Gabor, D., 1946. Theory of communications. *Journal of Institute of Electrical Engineering*, vol. 93, pp. 429-457.

[5] Phillips J.L. & Noelle, D.C. (2005). A biologically inspired working memory framework for robots. Proc. *of the 27th Annual Meeting of the Cognitive Science Society*, Stresa, Italy, July,

[6] Oliver, N.M., Rosario, B., and Pentland, A.P., “A Bayesian computer vision system for modeling human interactions,” *IEEE Trans. PAMI*, vol. 22, no. 8, pp. 831-843.

[7] Hongeng, S., Bremond, F., and Nevatia, R., “Bayesian framework for video surveillance application,” *Proc. ICPR 2004*, pp. 164-170, August, 2004.

[8] Wilkes, D.M., M. Tugcu, J.E. Hunter, and D. Noelle, "Working Memory and Perception", Proceedings of 14th Annual IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN 2005), Nashville, TN, August 13-15, 2005, pp 686-691, 2005.

[9] Markou, M. & Singh, S. (2003). Novelty detection: a review-part1: statistical approaches. *Signal Processing*, 83, 2481-2497.

[10] Di Stefano, L.; Mattoccia, S.; Mola, M., "A change-detection algorithm based on structure and colour," *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* , vol., no., pp. 252-259, 21-22 July 2003

[11] Xiaojing Yuan; Zehang Sun; Varol, Y.; Bebis, G., "A distributed visual surveillance system," *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* , vol., no., pp. 199-204, 21-22 July 2003

[12] Zehang Sun; Bebis, G.; Miller, R., "Boosting object detection using feature selection," *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* , vol., no., pp. 290-296, 21-22 July 2003

[13] Beynon, M.D.; Van Hook, D.J.; Seibert, M.; Peacock, A.; Dudgeon, D., "Detecting abandoned packages in a multi-camera video surveillance system," *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* , vol., no., pp. 221-228, 21-22 July 2003

[14] Bhargava, Medha; Chia-Chih Chen,; Ryoo, M. S.; Aggarwal, J. K., "Detection of abandoned objects in crowded environments," *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on* , vol., no., pp.271-276, 5-7 Sept. 2007

[15] Latecki, L.J.; Xiangdong Wen; Ghubade, N., "Detection of changes in surveillance videos," *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* , vol., no., pp. 237-242, 21-22 July 2003

[16] Sivic, J.; Russell, B.C.; Efros, A.A.; Zisserman, A.; Freeman, W.T., "Discovering objects and their location in images," *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on* , vol.1, no., pp. 370-377 Vol. 1, 17-21 Oct. 2005

[17] Micheloni, C.; Foresti, G.L., "Fast good features selection for wide area monitoring," *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* , vol., no., pp. 271-276, 21-22 July 2003

[18] Radke, R.J.; Andra, S.; Al-Kofahi, O.; Roysam, B., "Image change detection algorithms: a systematic survey," *Image Processing, IEEE Transactions on* , vol.14, no.3, pp. 294-307, March 2005

[19] Valera, M.; Velastin, S.A., "Intelligent distributed surveillance systems: a review," *Vision, Image and Signal Processing, IEE Proceedings* - , vol.152, no.2, pp. 192-204, 8 April 2005

[20] Durucan, E.; Ebrahimi, T., "Moving object detection between multiple and color images," *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.* , vol., no., pp. 243-251, 21-22 July 2003

[21] Bevilacqua, Alessandro; Vaccari, Stefano, "Real time detection of stopped vehicles in traffic scenes," *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on* , vol., no., pp.266-270, 5-7 Sept. 2007

[22] Jong Taek Lee,; Ryoo, M. S.; Riley, Matthew; Aggarwal, J. K., "Real-time detection of illegally parked vehicles using 1-D transformation," *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on* , vol., no., pp.254-259, 5-7 Sept. 2007

[23] Rosin, P., "Thresholding for change detection," *Computer Vision, 1998. Sixth International Conference on* , vol., no., pp.274-279, 4-7 Jan 1998

[24]Sijun Lu, Jian Zhang and David Dagan Feng, Detecting unattended packages through human activity recognition and object association, Pattern RecognitionVolume 40, Issue 8, , Part Special Issue on Visual Information Processing, August 2007, Pages 2173-2184.

[25] E. Durucan and T. Ebrahimi, Change detection and background extraction by linear algebra, Statistical change detection, *Proc. IEEE* **89** (2001) (10), pp. 1368–1381.

[26]L. Marcenaro, M. Ferrari, L. Marchesotti, C.S. Regazzoni, Multiple object tracking under heavy occlusion by using Kalman filters based on shape matching, in: Proceedings of International Conference on Image Processing, vol. 3, 2002, pp. 341–344.

[27] L.R. Rabiner, B.H. Juang, An introduction to hidden Markov models, IEEE ASSP Mag. (1986) 4–16.

[28]F.S. Hill Jr., Computer Graphics Using OpenGL (second ed.), Prentice-Hall, Englewood Cliffs, NJ (2001).

[29]M. Bosch, F. Heitz, J.P. Armspach, I. Namer, D. Gounot, and L. Rumbach, "Automatic change detection in multimodal serial MRI: Application to multiple sclerosis lesion evolution," Neuroimage, vol. 20, pp. 643-656, 2003.

[30]M.J. Dumskyj, S.J. Aldington, C.J. Dore, and E.M. Kohner, "The accurate assessment of changes in retinal vessel diameter using multiple frame electrocardiograph synchronised fundus photography," Current Eye Res.,vol. 15, no. 6, pp. 652-632, Jun. 1996.

[31]Shepard, R.N. (1964). Attention and the metric structure of the stimulus space. Journal of Mathematical Psychology, 1:54-87

[32] H. M. Kalayeh and D. A. Landgrebe, Predicting the Required Number of Training Samples, PAMI(5), No. 6, pp. 664-666, November 1983.