Robot Navigation and Textural Analysis

Rand C. Chandler rand@mil.ufl.edu
A. Antonio Arroyo arroyo@mil.ufl.edu
M. Nechyba nechyba@mil.ufl.edu
E. M. Schwartz ems@mil.ufl.edu

Machine Intelligence Laboratory
Department of Electrical and Computer Engineering
University of Florida, Gainesville, FL 32611-6200

Abstract

We present a method for navigating an autonomous agent based on the textures present in an environment. Specifically, the autonomous agent in question is that of a robotic lawn mower. If we can successfully differentiate the textures of the cut and uncut lawn surfaces, then we can track the boundary between them and mow in a pattern as a human would. The system uses the wavelet transform as the basis to perform texture analysis. The wavelet transform extracts meaningful features from the input images by breaking the image into different frequency subbands. Different subbands will isolate different features in the input image. In this way, we can generate a frequency signature of the image. After performing the wavelet transform, we perform a post-processing stage on these resulting features in an attempt to make them more acceptable to our classifier. These processed features are then grouped into vectors and then classified. The result is a clustered image based on texture. Once we have the image segmented based on the textures present in the image, we then determine the boundary between them by use of a boundary detection algorithm. In this way we can give a robotic lawn mower the ability to track this boundary and mow as a human would mow. While we avoid the actual implementation of this algorithm on a real platform due to the hazardous nature of lawn mowing in general, we do show how this algorithm can be easily adapted to the task of sidewalk tracking. In this alternate task, the robot tracks the boundary on both sides of the sidewalk, giving the robot the ability to follow the sidewalk. In doing so we not only show the adaptability of our algorithm to another task but also show its implemention on a mobile robot platform.

1. Introduction

Navigating a robot through its environment can be a daunting task depending on the degree of functionality we desire. For instance, we may want to design a robot that simply avoids (while moving randomly) bumping into obstacles in a closed (confined) indoor environment. On the other hand, we may desire a robot that operates outdoors and needs the ability to know exactly where it is in its environment. Both of these situations require that the autonomous agent have the ability to sense its environment for successful navigation tasks. This is typically accomplished through the judicious processing of sensory driven data.

The types of sensors used on an autonomous agent depends on the application. For our indoor robot case, simple infrared emitter and detector sensors may be all that is necessary to avoid bumping into obstacles. However, for a robot that needs to accurately ascertain its location outdoors, some type of positioning system is required. This may require the use of a Global Positioning System (GPS) or a Local Positioning System (LPS). A GPS based system consists of a GPS receiver that provides an approximate location based on timing information received from a constellation of orbiting satellites. A LPS can be realized through the use of beacons placed throughout the operating environment and the use of triangulation to determine position.

The reliance on an external positioning system could be lessened (or possibly eliminated) if the robot was given the ability to perceive its environment through the use of computer vision. If a robot could recognize objects in its environment, theoretically it should be able to navigate through it. Such a system could be used for a variety of applications. For example, an autonomous lawn mower could determine the difference between cut and uncut grass in a typical lawn and use this information to track the boundary between these two regions.

In order to show that a robot can navigate through its environment using computer vision, we develop a system which allows a robotic lawn mower to mow in a pattern similar to a human performing the same task. A human operator will typically mow a lawn using a plow-like (going back and forth) pattern or by starting at the outside perimeter of the mowing area and then mowing inward toward the center of the mowing area. Even though robotic lawn mowers do exist, they mainly rely on the principle of randomness to mow an area defined by a radio pet fence or other boundary defining system. The autonomous agent has no way of perceiving if it is mowing over a previously cut area or if it is mowing in an area that has not been cut. While this process does work, it is very inefficient in terms of time and energy consumption.

The system that we developed has the ability to recognize textures in an input image of a lawn allowing the mower to recognize the difference between the cut and uncut lawn surface. This gives the agent the potential to track the boundary between these two regions resulting in a net time savings and lower energy consumption. Because of the hazardous nature of lawn mowing, our algorithm was applied to a sequence of captured image files.

While the main emphasis of this paper focuses on the development of an agent capable of autonomous lawn mowing, we also show how the vision capable system can be adapted to other tasks. One such task is that of navigating a sidewalk. We then discuss how this can be extended to a real-time vision equipped agent.

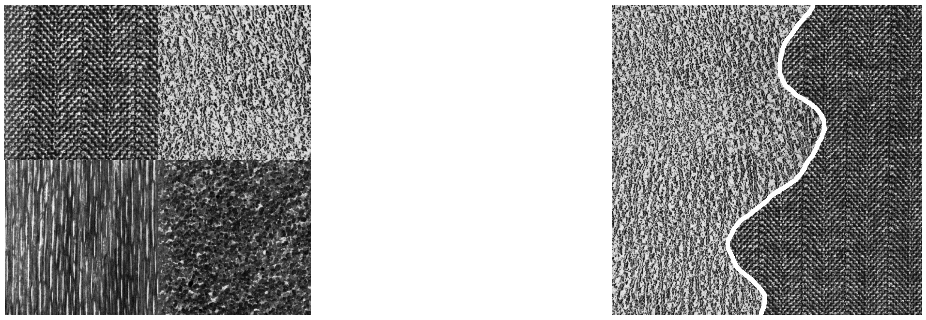

(a) (b)

Fig. 1: (a) Some sample Brodatz textures. From upper left going clockwise, D17: Herringbone weave, D24: Pressed calf leather, D68: Wood grain, D29: Beach sand. (b) An example of texture segmentation.

This system is also implemented on a non-mower mobile robot platform for the task of sidewalk tracking.

2. Metic of texture

Texture is an important quality to consider when examining the contents of an image. Practically every object (natural or man made) contains some texture on the macroscopic level. Thus, the use of texture information would be a practical means of segmenting objects in an image.

Figure 1(a) shows some examples of different textures. They are all from the Brodatz texture database. Figure 1(b) shows a sample result of texture segmentation. This is what an "ideal" segmentation would look like. Discussed in the sub-section below are the methods commonly used in the analysis of textures.

2.1 Methods used for texture analysis

Several methods have been used in the analysis of texture. These methods can be broken down into two main categories: (1) statistical based methods and (2) spatial/frequency based methods. Statistical based approaches include the use of Markov Random Fields [9, 12, 13, 16], co-occurrence matrices [14, 17], region competition [20], circular-mellin features [15], and second order grey level statistics [14]. Unser [17] reports that most statistical based methods are best suited for the analysis of micro textures. This is due to the fact that they are restricted to "the analysis of spatial interactions over relatively small neighborhoods." Spatial/frequency based approaches include the use of Gabor filters [2, 5, 10] and Wavelet transforms [6, 7, 8].

2.1.1 Spatial/frequency based approaches

These approaches include Gabor filters [2, 10] and wavelet transforms [6, 18]. Gabor functions consist of sinusoids of a particular frequency that are modulated by a Gaussian envelope. They (like their wavelet counterparts) provide localized space-frequency information of a signal [3]. Two dimensional Gabor functions consist of a sinusoidal plane of some orientation and frequency modulated by a Gaussian envelope. However, Gabor filters are computationally very inefficient [6]. Also, the outputs of Gabor filter banks are not mutually orthogonal which could result in correlation between texture features [17].

Wavelet transforms can be thought of as a multi-resolution decomposition (multiple scale signal view) of a signal into a set of independent spatially oriented frequency channels [1, 19]. In its simplest form, the wavelet can be thought of as a bandpass filter. To begin a wavelet analysis, one starts with a prototype (or mother) wavelet. High frequency (contracted) versions of the prototype wavelet are used for fine temporal analysis while low frequency (dilated) versions are used for frequency analysis [4]. Unlike the Gabor functions described earlier, fast algorithms exist for wavelet decompositions. If one chooses wavelets that are orthonormal, then correlation between different wavelet scales is avoided. Furthermore, there is experimental evidence that suggests that the human vision system supports the notion of a spatial/frequency multi-scale analysis, making wavelet transforms an ideal choice for texture analysis [8, 17]. For this project, texture will be represented using a spatial/frequency approach by use of the wavelet transform.

3. Texture analysis

The texture analysis algorithm can be logically broken into two distinct stages. In the first stage, a statistical model is created for the data contained within our training set. This stage is referred to as the training phase. For the second stage of this algorithm, the statistical model generated during the training phase is used to compare the contents of an unknown image against in order to classify which textures are present in the image. This stage-is referred to as the classification stage.

3.1 Training phase

Figure 2 shows a graphical representation of the training phase of our texture analysis system. As can be seen, the inputs to our system are composed of the various training data for the objects in which we are interested in identifying later in the classification stage. For this algorithm, we operate only on greyscale images. Thus, all color information is ignored. Features are extracted from these greyscale images, one image at a time, by means of the wavelet transform.

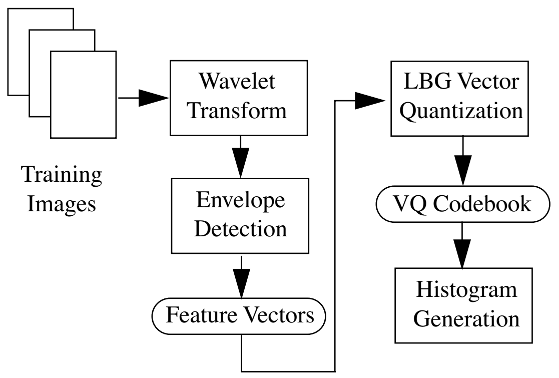


Fig. 2: Training phase of the texture analysis system.

3.1.1 Wavelet transform

Because we are dealing with images which are two-dimensional, we will carry out our analysis by means of a two-dimensional wavelet transform. Specifically, we will use the LeMarie-Battle wavelet/filter with an arbitrary wavelet decomposition of a discrete wavelet packet tree. Thus, not all sub-bands of a particular level will be used for the purposes of generating the VQ codebook which will later be used for classification.

3.1.2 Feature enhancement

After features have been extracted from a particular training image by means of the wavelet transform stage, the envelope detection stage is used to make the output of the wavelet transform stage more acceptable to the clustering and classification stages. We should note several important points when applying the envelope detector to a wavelet sub-band. In our case, the four distinct wavelet sub-bands generated as a result of the two-dimensional wavelet transform have different characteristics. For example, the LL sub-band is a down sampled version of the original image (or in a multiple level wavelet transform, a down sampled version of the previous node in the tree). Thus, application of the envelope detector to this band would not yield any significant results. However, the LH, HL, and HH bands do tend to isolate horizontal, vertical, and diagonal features respectively. It would therefore make sense to apply the envelope detector either column-wise or row-wise depending on the particular isolation properties of that particular wavelet sub-band. Given this is the case, we will apply the envelope detection algorithm according to Table 1.

Table 1: Orientation application of the envelope detector

Isolated Features	*Apply envelope det.*
Horizontal	column-wise
Vertical	row-wise
Diagonal	column-wise

3.1.3 Feature vectors

While the generation of feature vectors is not a process or stage in the overall process, it is important to discuss how the "feature" vectors are formed. The feature vectors are formed in the following manner: A vector is constructed for each pixel in the input image consisting of the values of the various wavelet sub-bands after the envelope detection process has taken place. It is important to point out that not all of the wavelet sub-bands for a given level will be used in construction of the feature vectors. This will be explained further in the next section. Also it is important to point out that the envelope detection stage is only applied to the last level in the wavelet tree. We assume that the envelope detection stage has been applied to the last stage in the wavelet tree before the vectors are formed from the wavelet subbands in that particular level.

3.1.4 Vector quantization stage

After the feature vectors have been generated for all of the training data, the LBG, vector quantization algorithm is applied to these vectors. What results is the VQ codebook. What we have not mentioned so far is what to choose for the size of the VQ codebook. We will defer this discussion until the next section.

3.1.5 Histogram generation

The histograms are generated for each class of training data. These are generated by counting the number of feature vectors that belong to each code in the VQ codebook for each object in the training set. The histograms are then normalized to fit probabilistic constranints. As mentioned earlier, because we are training the system with known data we have provided, we know from which object each feature vector came from. It is this knowledge we use to construct the histograms.

3.2 Clustering and classification phase

Now that we have the VQ codebook and histograms generated for our training data, we can now turn our attention to clustering and classifying unknown data. The unknown data is greyscale image data just as was the case for the training data. However, we will limit our unknown image size to 320 pixels horizontally by 240 pixels vertically. An image size of 320 by 240 pixels is large enough to maintain the textural features of the objects present in the image and is smaller than the traditional image size of 640 by 480 pixels. The reduced image size significantly reduces the computational time of the texture analysis algorithm. Figure 3 below shows a graphical representation of the clustering and classification stage.

It is important to point out that the wavelet transform, envelope detection, and feature vector generation stages are identical

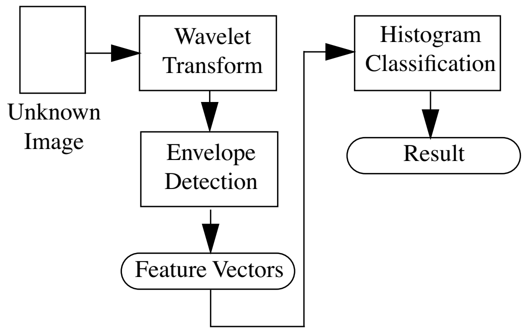


Fig. 3: Classification phase of the texture analysis system.

to that of the training phase. The same meaningful features that were extracted from the training data must be extracted from the unknown data in the same manner in order for our analysis to work properly. Once we have generated the feature vectors for our unknown data, we can then use the VQ codebook and find out which cluster (or code) the unknown vectors are closest to and then use the histogram data to make a determination to which class the unknown pixels belong. This stage is labeled "Histogram Classification" in Figure 3.

4. Lawn texture classification system

In this section, we show how the texture analysis system presented in the previous section is used to perform the task of differentiating between the cut and uncut surfaces of a lawn. Thus, if the mower knows where the boundary is between the cut and uncut lawn surfaces, it can track this boundary as it mows. In this manner, a system is produced that is capable of mowing similarly to how a human would mow a lawn. This is the main emphasis of this research. For the robot mowing task, lawn images are collected and processed offline (not in real time) to avoid the hazards associated with actual lawn mowers. In the next section we will show how this same system can easily be tailored to the task of tracking a sidewalk. Because of the non-hazardous nature of this task, this task is implemented on an autonomous robot.

4.1 Criteria for wavelet subband determination

In order to determine which wavelet sub-bands are relevant for the task of lawn texture classification, we need to have some type of criteria to judge each wavelet subband individually. Those that meet our criteria will be kept for use in the formation of the feature vectors. Listed below are the criteria we chose to determine if a particular wavelet subband is suitable for our purposes.

1. In order to keep the processing time down to a minimum, we would like to keep the number of wavelet subbands down to a minimum and still produce acceptable results. This also relates to which level of wavelet tree we generate since the number of nodes in the wavelet tree increases exponentially as one descends from one level to the next. Thus, we will limit our analysis to a level 2 wavelet tree producing 16 wavelet subbands.
2. Because of the strong vertical features exhibited by the lawn surface, we will disregard any HL sub-band (and any of its descendants) for analysis except if its parent was from a LH band. This HL band tends to isolate vertical features that are present in the uncut and (to some degree) the cut lawn surfaces, thus, this band is not useful to add to the content of the feature vectors. An LH band is beneficial due to the fact that this band isolates horizontal features which are not contained in the uncut lawn texture but are contained in the cut lawn texture. Thus this band (and any of its descendents) will be particular useful for our classification purposes.
3. From a mathematical perspective, we can compute the amount of energy contained within each wavelet subband. We will use Equation 1, known as the L1-norm or variance, to accomplish this [1]. In Equation 1, M and N represent the dimensions of the image. This equation sums up the absolute values of all the coefficients present in the particular wavelet subband. After that, the sum is then divided by the number of pixels in the image.

$$e = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}|x(m,n)| \tag{1}$$

4. For a final determination, we can visualize a particular wavelet sub-band by performing a list density plot. A mathematical package such as Mathematica or other similar tool can be used for this purpose.

4.2 VQ Parameter determination

Due to the self initializing nature of the LBG, VQ algorithm, only one parameter needs to be determined. This parameter is the number of codes in the VQ codebook. Too few codes or too many codes can lead to unsuccessful classifications. Also, because of the high dimensionality of our feature vectors, the larger the codebook, the longer it will take to generate the total codebook. Given these considerations, and experimental results, we will use a codebook size of 8 codes. Larger codebook sizes did not yield a significant improvement in the classification of the lawn textures.

4.3 Results

In this section results are shown for the lawn texture classification system. We begin our discussion by showing a sample lawn image and then we choose the appropriate wavelet subbands based on the criteria outlined earlier. Next, we describe how we obtain the training data for the system. After this, we present the clustered and classified results for several lawn images. We also discuss three algorithms for determining the boundary between the cut and uncut lawn surfaces.

4.4 Collection of lawn image data

To collect data for our analysis, a wheeled cart was constructed that allowed a standard camcorder to be mounted to it. A manual, gasoline powered, push mower was used to mow a strip of lawn in a typical lawn environment. Next the cart was pushed along the right edge of the path created by the push mower, thus capturing images consisting of the cut lawn surface on the left and uncut lawn surface on the right. After capturing the video, it was then input into a computer where it was isolated into individual image frames, downscaled to 320x240 pixels, cropped, and converted to greyscale. We crop the images vertically to get rid of the upper portion of the image. This has to do with the fact that the camera sits less than a foot off the ground and is slightly angled to the ground. Thus, the top of the image appears (and is) further away and does not lend itself to good classifications because the strong vertical features of the cut side are not as prevalent as they are in the lower portion of the image. Figure 4 shows a lawn image captured from our system. As one can see, due to the low height of the camera, the vertical features of the uncut portion the image are highly emphasized lending itself to a better classification.

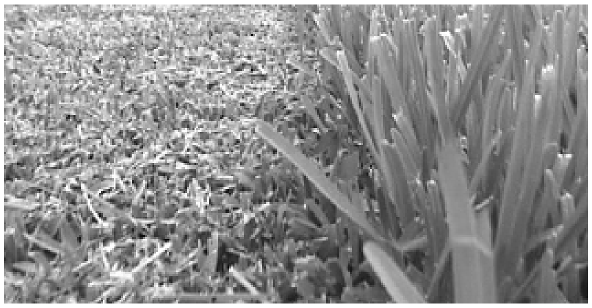


Fig. 4: Captured lawn image. The cut lawn surface is on the left. The uncut lawn surface is on the right.

4.5 Determination of wavelet subbands for lawn texture analysis

Figure 5 shows the list density plots for each of the 16 wavelet subbands that are generated in the level 2 DWPF wavelet tree decomposition. In Figure 5, below each subband is text describing what type of subband it is (LL, LH, HL, HH) and from what type of subband it is descended from. The number at the end is the value of the L1, norm energy measure as described in Equation 1. It is important to note that this energy measurement is computed before the application of the envelope detection stage.

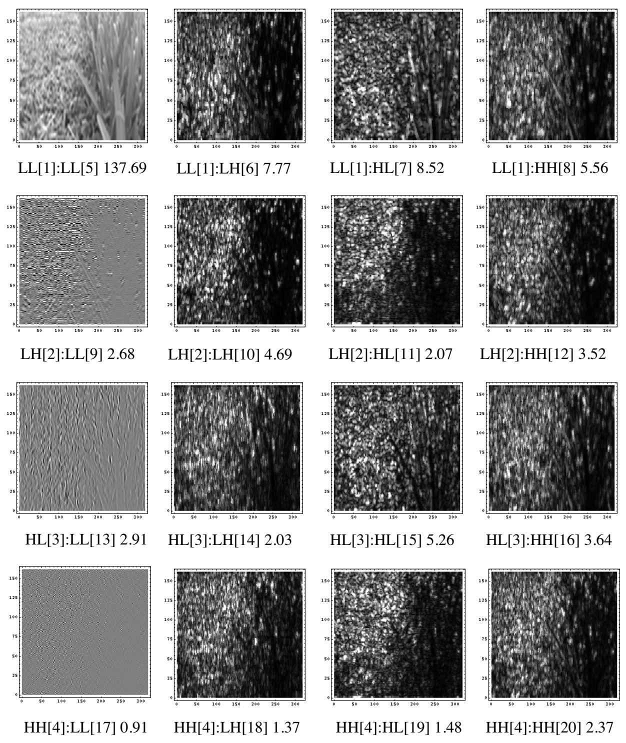


Fig. 5: Wavelet sub-bands contained in level 2 of the DWPF for the image shown in Figure 4. The text, XX[A]:YY[B] C, below each subband can be interpreted as follows: XX represents the parent node from which this sub-band is generated, YY represents what type of subband this is, A and B represent the index number for each subband, and C represents the L1, norm energy measurement for each subband.

Now that we have some information from which to make an assessment, let us use the criterion we listed previously to justify our selection of the wavelet subbands used for the lawn texture classification system. First, let us state the subbands that we do intend to use for analysis purposes and then explain why we eliminated the others. We will use subbands 6, 8, 10, 11, and 12.

Now we will describe how we eliminated the other subbands. First of all, we will eliminate subbands 13 through 16 because they are descendants of the top level HL[3] band. Secondly, we will eliminate bands 17 through 19 based on their low energy content. Although band 20 may be useful for classification purposes, we eliminate this band to keep the number of subbands to a minimum. Next, we eliminate band 5 because this is a lowpassed version of the original image. Band 7 gets eliminated because it is a HL band which is descended from the LL[1] band which is a lowpass version of the original greyscale image. Finally, we eliminate band 9 since it is a LL band.

4.6 Selection of the training data

Now that we have determined which wavelet subbands will be used for the purposes of lawn texture analysis, we need to obtain training data in order to generate the VQ codebook. This codebook, as stated in the previous section, is generated during the training phase. To obtain training data, we cropped small samples, 2 for the cut lawn surface and 2 for the uncut lawn surface from one frame of image data. This was the only training data used for the entire series of lawn images that were classified. Figure 6 shows the training data used to generate the VQ codebook.

4.7 Clustered and classified results

Below we present several images showing the results of our texture analysis system for determining the boundary between the

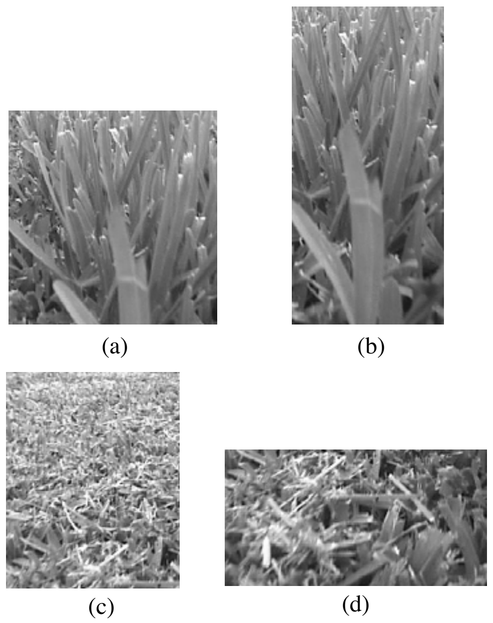


Fig. 6: Training data used to generate the VQ codebook for our results. Figures (a) and (b) represent the cut training data and figures (c) and (d) represent the uncut training data

cut and uncut lawn surfaces. We will also discuss three separate methods used to calculate the linear boundary between the cut and uncut lawn regions in each image. Figure 7 shows some results of our system.

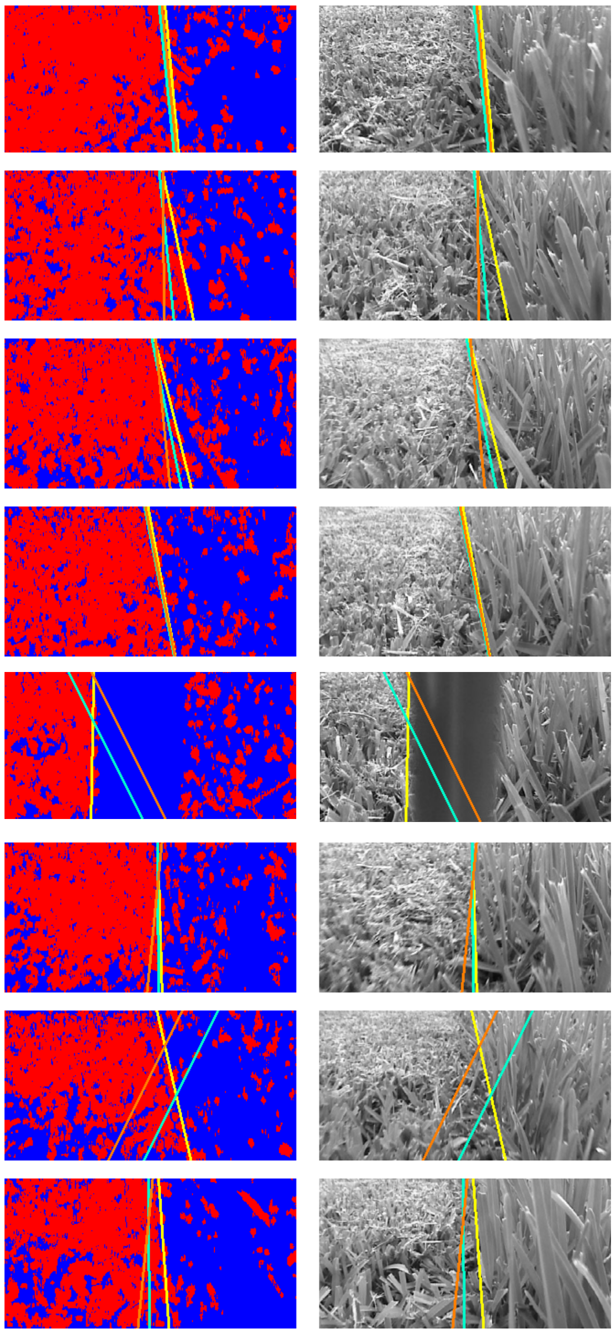


Fig. 7: Some results of the lawn texture analysis system.

4.8 Line boundary determination

Three methods were employed to determine the linear boundary between the cut and uncut classified regions. If the mower is supposed to mow as a human would mow, it needs to be able to determine the boundary between the cut and uncut regions. What follows is a description of the three methods used.

4.8.1 Best fit step method

This first method is performed on a line by line basis by attempting to fit the data of the two classes to the best fit (lowest least-squared error) step function [11]. This is depicted graphically in Figure 8 below. The parameter $d(i, j)$, is referred to as the discriminant and in our case is simply the value given to each class. For instance, $d(i, j) = 1$ for the pixels classified as the cut lawn texture and $d(i, j) = 0$ for pixels classified as the uncut lawn texture.

To compute the best fit step function we exhaustively compute the least squared error function (shown in Equation 2) for each jd in line i. This involves computing the means ml and mr which represent the mean of the data on the left side of jd and the mean of the data on the right hand side of jd respectively. The value of jd which yields the lowest error is assigned to be the best fit for that particular line.

$$error = \frac{\sqrt{\sum_{j=0}^{jd}[d(i,j)-ml]^2 + \sum_{jd+1}^{jmax}[d(i,j)-mr]^2}}{jmax+1} \tag{2}$$

Once we find the best fit step for every line in the image, we use linear regression to plot a line through the best fit step points. To take care of any points that may be outliers, after performing linear regression, we go back and calculate the distance from each best fit step point and the line itself. A point that fits within our prescribed tolerance is kept, all other points are discarded. After this, we perform linear regression again using only these acceptable points thus producing a line that better fits the given data points. In Figure 7, this line is represented by the yellow line.

4.8.2 Maximizing the Minimum Method

In this method, we begin by partitioning the classified result into two parts by separating them linearly by a candidate line. We refer to the line as a candidate line since we will try several lines and see which one gives us the least error according to this method. The line that yields the least error will be the boundary between the cut and uncut lawn textures. The algorithm is outlined below:

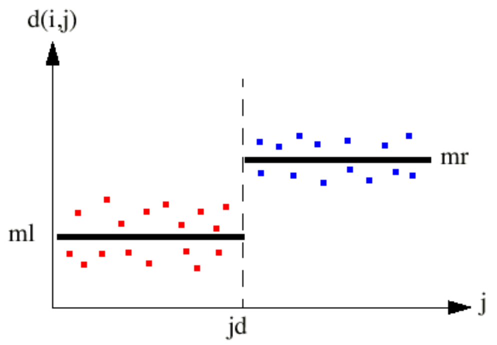


Fig. 8: Graphical representation of performing the best fit step function method.

1. Set $max = 0$.
2. Linearly partition the classified result into two by the candidate line.
3. Let *p1* be the percentage of pixels from class 1 on the left hand side of the candidate line and let *p2* be the percentage of pixels from class 2 on the right hand side of the candidate line.
4. If $p1 < p2$ then $min = p1$, else $min = p2$.
5. If $min > max$ then let $max = min$.
6. Go to Step 2 until the classified result has been partitioned by all possible candidate lines.

The candidate line where the condition in Step 5 of the algorithm was last met is the one chosen to be the boundary between the cut and uncut lawn surface. This line is represented in Figure 7 by an orange line.

4.8.3 Maximization of area method

This method is similar to the "maximizing the minimum" method in that we partition the classified result into two parts by the selection of an appropriate candidate line. In this method, we try to linearly partition the boundary between the two classes in such a manner that maximizes the area of the two classes on either side of the line. For instance, the best possible line would be one where all the pixels from one class were on one side of the line and pixels from the other class were on the other side of the line. Obviously we may never have an ideal case, but we are interested in finding the line which maximizes this criterion. We formally state the algorithm below:

1. Set $maxavg = 0$, $min = 1$.
2. Linearly partition the classified result into two by the candidate line.
3. Let *p1* be the percentage of pixels from class 1 on the left hand side of the candidate line and let *p2* be the percentage of pixels from class 2 on the right hand side of the candidate line.
4. Let $diff = fabs(p1 - p2)$. If $diff \leq min$ then $min = diff$ and $avg = (p1 + p2)/2$.
5. If $avg > maxavg$ then $maxavg = avg$.
6. Go to Step 2 until the classified result has been partitioned by all possible candidate lines.

The candidate line where the condition in Step 5 of the algorithm was last met is the one chosen to be the boundary between the cut and uncut lawn surface. This line is represented in Figure 7 by an aqua line.

4.9 Line boundary methods conclusions

The results of all three methods used for determining the boundary between the cut and uncut lawn surface can be seen in Figure 7. As can be seen, for the most part, all three methods tend to produce results which are similar to one another. However, sometimes the maximizing the minimum and maximization of area methods do not yield an appropriate result. This is a result of the general nature of these methods. Both of these methods rely on the fact that pixels from one class should be present on one side of the line and that pixels from the other class be present on the other side of the line. When there are patches of mis-classification, this can have a detrimental effect on these two algorithms. The best fit method can overcome this due to the fact that the line is determined by a series of points. Even if some points are not correct, the boundary can still be determined with reasonable success.

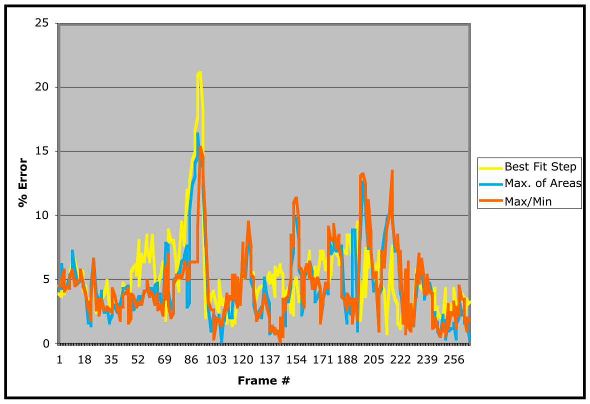


Fig. 9: Graph of the calculated error measurements for each line boundary detection method for consecutive image frames.

To gain further insight into the effectiveness of these line boundary methods, we calculate an error measurement for each of the described methods. This is accomplished by comparing the line generated by each method to a line that has been generated through visual inspection by a human. Thus, this serves as a basis to judge the efficacy of our described methods.

To determine the error between a line generated by one of our methods and the line generated by visual inspection, we calculate the number of pixels between the two lines and then divide this number by the total number of pixels present in the image to get a percentage of the pixels contained between the two lines. If the lines happen to intersect one another (forming two triangular regions) then we count the number of pixels contained within these two triangles and then divide by the total number of pixels in the image. Thus, the lower the error, the closer the line generated by one of the boundary detection methods and line determined by visual inspection should be to one another.

To illustrate this, the boundary line was determined by visual inspection for 266 consecutive frames. The boundary line was also computed using the three line boundary methods. The error was then computed for every frame for each line boundary method. Figure 9 shows a plot of the error measurements for each of the line boundary methods for the 266 consecutive frames. The large spike in error centered around frame 86 is due to the occlusion of the image by a single blade of glass. This can be seen in one of the output frames in Figure 7.

5. Sidewalk texture classification system

In this section, we adapt the texture analysis algorithm to perform sidewalk identification. This in turn, will be used to allow an autonomous mobile robot to follow a sidewalk as it moves forward. A system which is designed to operate an autonomous robot should operate close to real time as possible. Otherwise, the robot may miss pertinent information necessary to carry out its task.

5.1 Criteria for wavelet subband determination

As was the case for our lawn texture classification system, it is necessary to chose appropriate wavelet subbands for our classification purposes. Instead of lawn textures, which consisted of the cut and uncut lawn surfaces, we will be dealing with textures such as concrete, asphalt, and some type of ground cover on the periphery of the sidewalk. Thus, we must consider different criteria for determining which wavelet subbands to use. Listed below are the following criteria.

1. Because this is a real-time application, we want to limit our analysis to as few wavelet subbands as possible.
2. We will use the same energy metric as was used in the previous chapter for computing the amount of energy (indicator of frequency content) in a wavelet subband. Those bands that posses high frequency content will be considered for analysis.
3. Finally, we will visually examine each subband by generating list density plots.

5.2 VQ parameter determination

As stated in the previous chapter, the only parameter we must specify in the VQ algorithm is the codebook size. Because of the real-time nature of this application, we will use a codebook size of 4 for our analysis.

5.3 Robot platform and system

To test our algorithm, we modified an existing robot platform to make it suitable for this task. The platform in question is a four wheeled robot in which the two wheels in the back of the robot are the drive wheels and a linear actuator type steering controller is used to control the front two wheels. In order to obtain images, a camcorder was mounted in a wooden frame that allowed the camera to be adjusted in terms of height and camera angle. This frame was attached to the front of the robot. A picture of the robot is shown in Figure 10.

To acquire image data from the camcorder, a PCI based video frame grabber card was used. In addition, we required a considerably fast system to run the texture analysis algorithm. These two requirements led to the decision to use a desktop PC. Specifically, we used an Intel P4 - 1.4GHZ computer with 256MB of RAM running the LINUX Mandrake 8.0 operating system. Because of this, the PC was placed on a cart that was pushed behind the robot. This can be seen in Figure 6-1. On top of the PC (as can be seen in Figure 10) a laptop computer was used to remotely login to the PC so that the program could be started and stopped and the results of the texture analysis algorithm could be viewed. To control the speed and direction of the robot, commands were sent to the robot via a serial cable connected to the PC. Finally, to provide power to the system, a small, portable generator was used. A graphical breakdown of the system is shown in Figure 11.

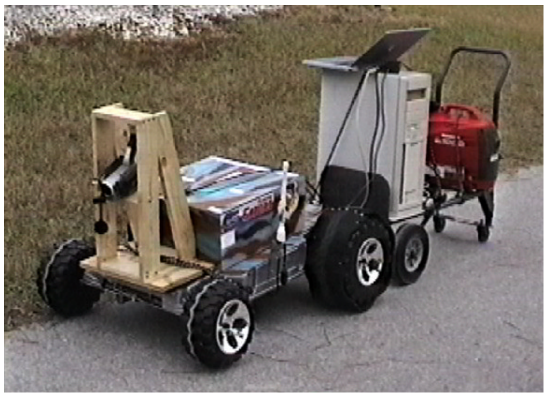


Fig. 10: Picture of the robot platform and support equipment.

5.4 Determination of wavelet subbands for sidewalk texture analysis

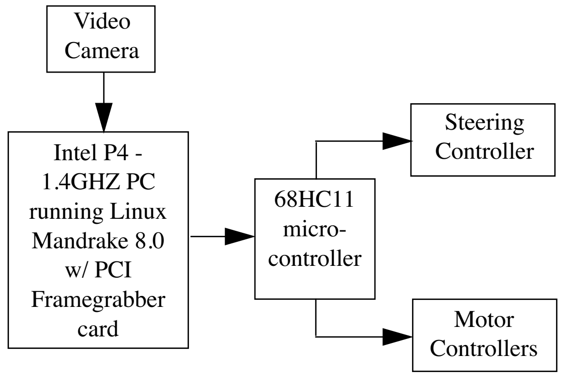
As in the previous chapter, we present in Figure 12, the list density plots for each of the 16 wavelet subbands that are generated in the level 2 DWPF wavelet tree decomposition of a sample sidewalk image. Below each sub-band is text describing what type of sub-band it is (LL, LH, HL, HH) and from what type of subband it is descended from. The number at the end is the value of the L1, norm energy measure as described in Equation 1. It is important to note that this energy measurement is computed before the application of the envelope detection stage.

As can be seen in Figure 12, subband 5, the second level low passed version of the original image, yields the highest distinction between the sidewalk and the periphery area. Thus, it will be included in our analysis. It should be noted that we did not consider this particular subband for our lawn texture classification system. For that application, this band did not yield a great distinction between the cut and uncut lawn surfaces. Also, the height and angle of the camera were different in the lawn classification application making the textures of the lawn surface more pronounced. With our sidewalk system, the camera is much higher off the ground, thus limiting the amount of textural information we can view from the objects in the image.

We will also include subbands 6, 8, and 10 for use in our analysis primarily based on the energy metric. To summarize, we will use subbands 5, 6, 8, and 10 for our sidewalk texture classification system.

5.5 Training data

Shown in Figure 13 below is the training data set used in all the experiments conducted with this system. The training data


Video Camera
Intel P4 - 1.4GHZ PC running Linux Mandrake 8.0 w/ PCI Framegrabber card
68HC11 micro-controller
Steering Controller
Motor Controllers
Fig. 11: Graphical representation of the system used for autonomous sidewalk navigation.

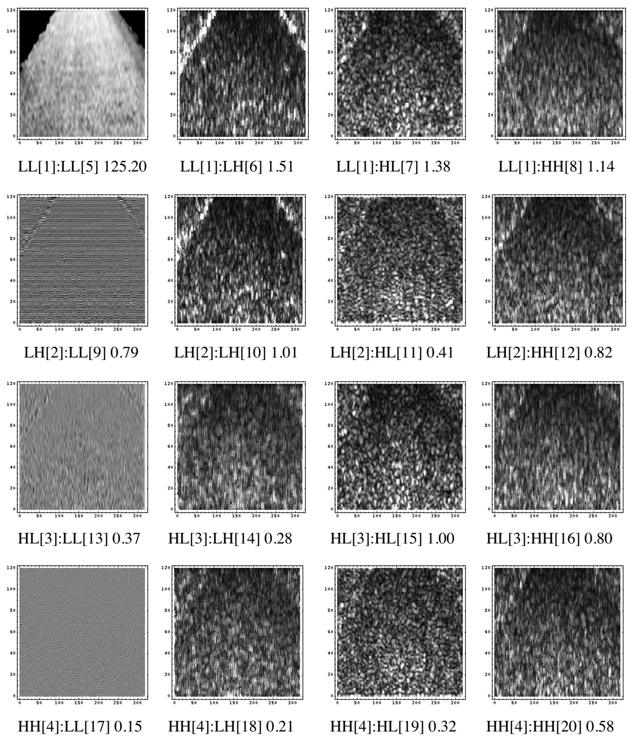


Fig. 12: Wavelet sub-bands contained in level 2 of the DWPF for a sidewalk image. The text, XX[A]:YY[B] C, below each subband can be interpreted as follows: XX represents the parent node from which this sub-band is generated, YY represents what type of subband this is, A and B represent the index number for each subband, and C represents the L1, norm energy measurement for each subband.

was acquired by capturing a single frame of video after the robot system was setup. We then cropped two regions of texture to form the training set.

5.6 Boundary Detection

As was the case for the lawn texture classification system where we needed to determine the boundary between the cut and uncut regions of the lawn surface, we need to determine the boundary between the sidewalk and periphery areas of the sidewalk. In this case, we must find the boundary on the left hand side and on the right hand side. To accomplish this, we divide the image into two halves vertically and then use the best fit step method to detect the boundary in each half. Once we have located both boundaries, we then find where the lines intersect the top of the image. Knowing this, we calculate the center of the intersection of the two lines. This will be the center of the sidewalk.

We have so far described an ideal situation in which we are in the center of the sidewalk and have a left and right boundary. However, if for some reason the robot veers to far off course, only one boundary line will be present. In order to deal with this case, we must constantly check the condition of the two lines. Under normal circumstances, one line should have a positive slope, the other negative. If only one line is present in the entire image, the slope of both lines should be the same polarity, namely either positive, positive or negative, negative. If this is the case, we then know we are in a situation where we have veered off course and are only seeing one boundary line. To get a better estimate of the line, we compute the best fit step for the entire image.

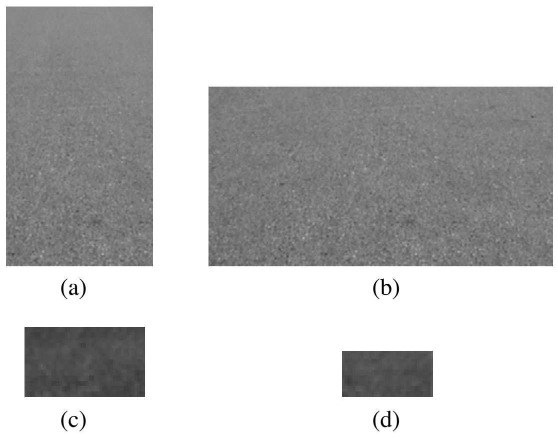


Fig. 13: Training data set for the sidewalk texture analysis system. Figures (a) and (b) show the training data set for the sidewalk texture and figures (c) and (d) show the training data set for the periphery texture.

5.7 Results

Now that we have discussed the robot platform, choice of parameters, and training data for our texture analysis algorithm, we now present some results of our system. We conducted 7 runs of our system. Our test area was an asphalt sidewalk surrounded by thinly bladed grass. The speed of the robot was set to the lowest speed possible (about 1 foot per second) and the algorithm processed roughly 3 frames per second. The area where the tests occurred was at a slight incline which aided in testing. Going up the incline had the effect of slowing the robot down to about .5 foot per second. This enabled the robot to process more frames per foot as the robot traveled making its tracking more accurate. Correspondingly, when going down the hill, the robot moved at about twice the speed as it would on a level surface. Because of this, the robot tended to oscillate from side to side as it traversed down the sidewalk. It is important to keep in mind that the robot was able to track the sidewalk as it moved in each of the scenarios. The robots steering was completely controlled by the algorithm, thus it was moving autonomously along the sidewalk. Figure 14 and Figure 15 show some sample results of our system.

5.8 Analysis of Results

As can be seen in Figure 14 and Figure 15, the robot tracks the sidewalk to a high degree of accuracy. The pictures contained in Figure 14 were collected from an uphill run of the robot. During this run, the robot stayed on course very well and oscillation from side to side was a minimum. This was due to the fact that the robot was slowed down enough to where it could keep up to the speed of the algorithm. However, as we can see in the pictures contained in Figure 15 where the robot travels at a faster rate downhill, the robot tends to oscillate from side to side.

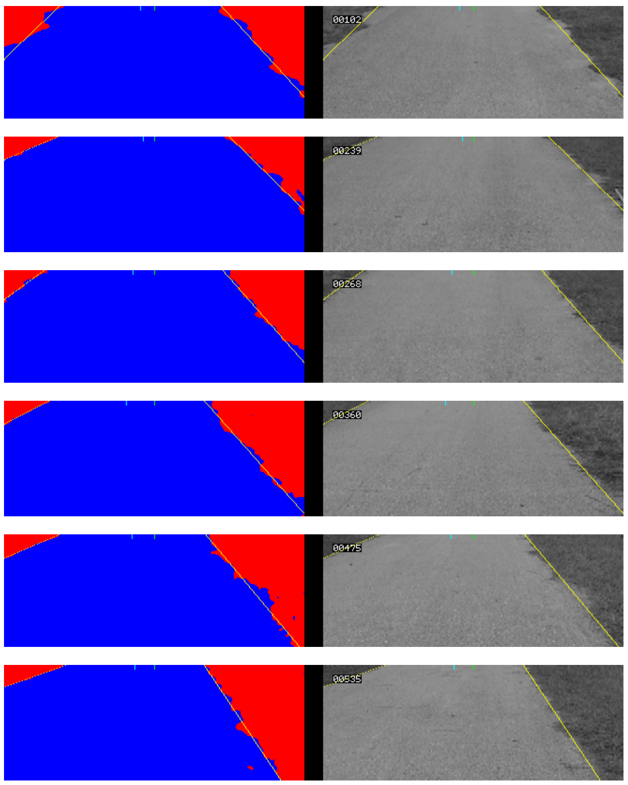


Fig. 14: Output results for our sidewalk texture classification system. The images on the left hand side represent the classified and clustered result of the images presented on the right hand side. These images are taken going up the hill.

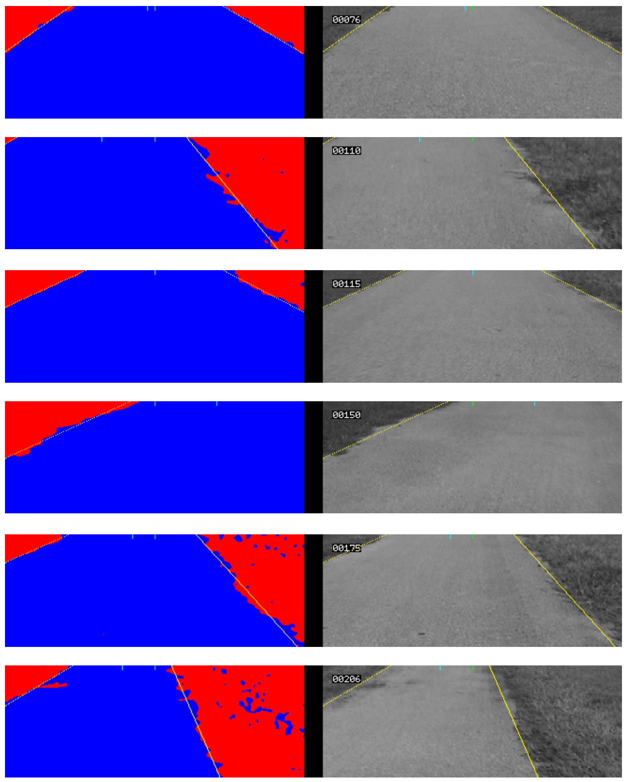


Fig. 15: Output results for our sidewalk texture classification system. The images on the left hand side represent the classified and clustered result of the images presented on the right hand side. These images are taken going down the hill.

6. Conclusions

Our main conclusion, as is the premise of this paper, is that robots can indeed navigate outdoor environments through the judicious use of computer vision. Specifically, we acquired images and used our texture analysis algorithm to identify specific attributes in these images. This was demonstrated by our lawn texture analysis system where we were able to successfully determine the difference between the cut and uncut lawn surfaces. In this way, we could determine the boundary between these two surfaces and use this to allow an autonomous lawn mower to mow in a pattern. We further demonstrated our claim by adapting our algorithm for the alternate application of sidewalk tracking. In this case, however, we implemented this on a mobile robot platform. That robot was successfully able to track a sidewalk as it moved along it. Several successful runs were performed on this system.

The texture analysis algorithm developed is a highly flexible one with few parameters to specify. As demonstrated by our sidewalk tracking application, this system can easily be adapted to other tasks as well. All that is required is the choice of the appropriate wavelet subbands and the choice of the codebook size for the vector quantization algorithm.

From our experiments, we can draw several conclusions. First of all, the use of texture analysis for image segmentation allows for greater flexibility of the environmental conditions in an outdoor environment. One of the greatest concerns for computer vision in an outdoor environment is the changing lighting conditions due to clouds, going under trees, etc.. Because we are identifying objects based on the physical property of texture, this system is more robust to changes in lighting conditions as wouldn't necessarily be the case if we were identifying the objects based on their color. In addition, for our lawn application, color would not be an applicable discriminant to use because we are trying to differentiate the same lawn surface (i.e. grass) in two different forms (cut and uncut).We can also judge the flexibility of our system based on the training of the system. For all our lawn and sidewalk experiments, we trained the system only once. The system showed remarkable flexibility in being able to successfully segment the textures in the images for long experimental runs. For the sidewalk tracking experiments, all 7 runs were performed using the same training data set. Secondly, we have demonstrated that the wavelet transform provides a valuable tool for extracting textural features from images. While much research has gone into the use of wavelet transforms for textural analysis, no one has applied this research to the area of robotics. The use of the wavelet transform was mainly confined to the task of analyzing samples of textures or samples of micro-textures in a laboratory environment. This is the first application where texture analysis has been applied to a real agent.

7. References

[1] K. W. Abyoto, S. J. Wirdjosoedirdjo, and T. Watanabe,"Unsupervised Texture Segmentation Using Multiresolution Analysis for Feature Extraction," 東京情報大学研究論集, Vol. 2, No. 1, pp 49-61, 1998.

[2] D. Dunn and W. E. Higgins, "Optimal Gabor Filters for Texture Segmentation," *IEEE Transactions on Image Processing*, Vol. 4, No. 7, pp. 947-964, Jul. 1995.

[3] F. Espinal, T. Huntsberger, B. Jawerth, and T. Kubota, "Wavelet-Based Fractal Signature Analysis for Automatic Target Recognition," *Optical Engineering, Special Section on Advances in Patten Recognition*, Vol. 37, No. 1, pp. 166-174, 1998.

[4] A. Graps, "An Introduction to Wavelets," *IEEE Computational Science and Engineering*, Vol. 2, No. 2, pp. 1-18, 1995.

[5] A. K. Jain and F. Farrokhnia, "Unsupervised Texture Segmentation Using Gabor Filters," *Pattern Recognition,* Vol. 24, No. 12, pp. 1167-1186, 1991.

[6] A. Laine and J. Fan, "An Adaptive Approach for Texture Segmentation by Multi-channel Wavelet Frames," *Center for Computer Vision and Visualization,* Technical Report No. TR-93-025, Computer and Information Sciences Department, University of Florida, Aug. 1993.

[7] A. Laine and J. Fan, "Frame Representations for Texture Segmentation," *IEEE Transcations on Image Processing*, Vol. 5, No. 5, pp. 771-779, May 1996.

[8] A. Laine and J. Fan, "Texture Classification by Wavelet Packet Signatures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 11, pp. 1185-1191, Nov. 1993.

[9] C-T Li and R. Wilson, "Textured Image Segmentation Using Multiresolution Markov Random Fields," *IEE Colloquium on Applied Statistical Pattern Recognition*, pp. 8/1-8/6, 1999.

[10] D. P. Mital, "Texture Segmentation Using Gabor Filters," *IEEE Fourth International Conference on Knowledge-Based Intelligent Engineering Systems & Allied Technologies*, pp. 109-112, Aug. 30 - Sept. 1, 2000.

[11] M. Ollis, "Perception Algorithms for a Harvesting Robot," PhD Dissertation, The Robotics Institute, Carnegie Mellon University, Aug. 1997.

[12] W. Pieczynski and A-N Tebbache, "Pairwise Markov Random Field and its Application in Textured Images Segmentation," *Proceedings of the 4th IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 106-110, Austin, Texas, 2000.

[13] G. Poggi and A. R. P. Ragozini, "Image Segmentation by Tree-Structured Markov Random Fields," *IEEE Signal Processing Letters*, Vol. 6, No. 7, pp. 155-157, July 1999.

[14] T. Randen and J. H. Husoy, "Filtering for Texture Classification: A Comparative Study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 4, pp. 291-310, Apr. 1999.

[15] G. Ravichandran and M. M. Trivedi, "Circular-Mellin Features for Texture Segmentation," *IEEE Transactions on Image Processing,* Vol. 4, No. 12, pp. 1629-1640, Dec. 1995.

[16] A. Sarkar, M. K. Biswas, and K. M. S. Sharma, "A Simple Unsupervised MRF Model Based Image Segmentation Approach," *IEEE Transactions on Image Processing*, Vol. 9, No. 5, pp. 801-812, May 2000.

[17] M. Unser, "Texture Classification and Segmentation Using Wavelet Frames," *IEEE Transactions on Image Processing*, Vol. 4, No. 11, pp. 1549-1560, Nov. 1995.

[18] M. Unser, "Texture Discrimination Using Wavelets," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 640-641, 1993.

[19] M. Vetterli, "Wavelets and Filter Banks: Theory and Design," *IEEE Transactions on Signal Processing*," Vol. 40, No. 9, pp. 2207-2232, Sep. 1992.

[20] S. C. Zhu, "Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multi-band Segmentation," *Harvard Robotics Laboratory*, Technical Report No. 94-10, pp. 1-50.