A Computer Vision System for Monitoring Production of Fast Food

Richard Russo　　Mubarak Shah　　Niels Lobo
Computer Vision Lab
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816
{rrusso, shah, niels}@cs.ucf.edu

Abstract

A system is presented for monitoring the work of a fast-food employee using a single static camera. Skin detection, shadow detection, and region-growing are used as low-level techniques in order to analyze a video of an employee preparing a hamburger or sandwich. At the end of the video, the system produces a description of the sandwich that was prepared, for example: `[bread, turkey, lettuce, bread]`. *This description gives the ingredients used in the sandwich as well as their physical arrangement. This information could be compared to company specifications in order to determine if the sandwich was produced correctly.*

1. Introduction and Motivation

In the spirit of the general problem of understanding video sequences, this paper presents a system to analyze sequences of human subjects preparing sandwiches. The goal for the system is to determine what kind of food is prepared. The system operates on an abstraction of the sandwich preparation situation; it monitors how an employee interacts with objects (items of food) initially stored in predefined locations. A video clip depicting these objects being arranged is analyzed, and the system determines what this arrangement is.

An application motivation for this system is monitoring fast-food production. As well as an experiment in computer vision, this system serves as a prototype for a system that could monitor fast-food employees preparing food. We will use some terminology in this paper reflecting this motivation, such as referring to the human subject of the video clips as an "employee."

2. Related Work

There is a large body of work on the analysis of human motion reported in the literature. Please see two excellent surveys [1] and [3] for a detailed treatment of this subject. For a sample of recent work refer to the special section of IEEE PAMI on Video Surveillance [2]. In the following, we briefly describe some sample work in this area, which is in no way exhaustive and complete.

Bobick and Davis [9] describe a method to recognize aerobic exercise in video sequences. First they apply change detection to identify moving pixels in each image of a sequence. Then MHI (Motion History Images) and MRI (Motion Recency Images) are generated. A MHI is basically the union of all images detected as changing, representing all the pixels which have changed in a sequence. A MRI is a scalar-valued image where more recent moving pixels are brighter. In their system, MHI and MRI templates are used to recognize motion actions (18 aerobic exercises). Several moments of a region in these templates are employed in the recognition process. The templates for each exercise are generated using multiple views of a person performing the exercises. However, it is shown that during recognition only one or two views are sufficient to get reasonable results.

Intille and Bobick [5] and Intille, *et al.* [6] discuss the use of context to enhance Computer Vision applications. In these articles, context is taken advantage of primarily to perform tracking.

The main goal of Rosin and Ellis's [4] system was to differentiate between humans and animals to reduce false alarms. This system is especially interesting because of its use of context to help improve recognition. To improve the performance of their intruder detection system, the authors include a map of the scene which shows areas such as sky, fence, and ground. This helps to differentiate between a person and an animal (such as a bird) moving through the scene.

Kojima *et al.* [7] propose an approach to generate natural language descriptions of human behavior from real video sequences. First, a head region of a human is extracted from each frame. Then, using a model-based method, 3-D pose and position of head are estimated. Next, the trajectory of the head is divided into segments, and the most suitable verb is selected.

Like many of the systems above, the system presented here uses motion, context, etc. to detect and classify events in a video sequence. However, the goal in the construction of this system is to synthesize the knowledge of the actions performed in the sequence to determine how a task is carried out.

3. Input

In this section, we qualify the kind of sequences that are appropriate for input to the system. The algorithm employed by the system relies on some a priori knowledge about the sequence. The configuration process is also described in detail in this section.

3.1. Input Sequences

The sequences that the system processes are of an abstraction of a fast-food situation. The main features of the scene are an employee, a workspace where food items are arranged, and several *food bins*, representing the stationary containers where food items are stored. In a typical fast-food restaurant, these food bins are built into the work area, and do not move around (they might be removed for cleaning).

The sequences described in this paper are filmed by a stationary camera, facing the employee. The workspace is approximately centered in the camera's view, between the employee and the camera. The food bins are arranged on the side of the workspace that is opposite the employee.

The system described in this paper makes use of some color techniques. Therefore, two restrictions are put on the colors of the scene. First, the color of the skin of the employee and the colors of the workspace and food items are disjoint. Second, the workspace is a solid color. This last requirement is justified in that a typical counter top where food is produced will have a plain, solid, color.

Finally, the employee should only use one arm at a time, and the arms should not join in a sequence. Analyzing interactions between the two arms becomes complex. The system uses a skin-detection technique to find the arms of the employee in the image. If the arms become merged, they look like one region to a system using this kind of technique. At this time, the authors chose to work only with sequences that avoid the merging problem, to permit work in other areas.

3.2. Configuration

The system requires some configuration before processing a sequence. The first item of configuration is a color predicate trained for the skin of the employee in the sequence. The second is a copy of the first image in the sequence, with the food bins marked by a sentinel color (pure green, for instance). The arms are also marked by a (different) sentinel color. This image should show the workspace without any foreign objects occluding it, including the employee's hands. This region is referred to as the *true workspace*, and the location of a single point in this workspace region is given to the system as configuration information. The last part of the configuration is the names assigned to the food items of the sequence's food bins.

Figure 1: Sample configuration image (for sequence 3).

The system processes the configuration image as an initialization step. Since the food bins are stationary, the system uses the bins marked in the configuration image to determine when an employee reaches into one of them. The true workspace is determined from the configuration image using the supplied starting point and a region-growing operation (see section 4.1.2). Searches for food and shadow are later confined to this true workspace region. The names assigned to the food bins are used by the system to create appropriate output. Knowing that a food item came out of certain bin, the system can use the specified name when referring to the item.

4 System Description

The fundamental aim of the system is to determine how the employee has arranged food items to create a sandwich. In doing this, the subtasks are partitioned into two levels. Low-level vision techniques are applied to the frames of the sequence to segment out the basic features: the arms of the employee, the workspace, and shadows. The output of these procedures is used at a higher level to determine what is happening in a particular frame.

4.1. Low-Level Module

In order to segment arms, workspace, and shadow out of an image, the system employs three low-level color-based vision techniques. A variant on the color predicate technique is used to detect the employee's arms by recognizing the skin color. Color-based region growing is used to determine the full region that the workspace occupies. Finally, a novel shadow detection technique is used.

4.1.1 Skin Detection

Skin detection is used to find the arms of the employee in the sequence. This is based on the color predicate technique, as described in [10].

A slight variation of this technique is used to improve accuracy in the presence of skin-colored objects in the scene that are not skin. We apply a color predicate to each image in a sequence, but only consider a subset of the results. We take the region detected as the arms of the previous frame (or the arms selected in the configuration image, for the first frame) A, and union this with the regions of the image that have changed from the last frame C. The result is a subset $S = A \cup C$ of the image that contains the best candidates for the arms in the image. Inside this subset, the two largest 8-connected components are taken to be the arms. The left arm is differentiated from the right arm by comparing the leftmost points of the two regions.

4.1.2 Region Growing

Since the workspace region is a solid color, a color-based region growing technique is applied to detect it. This region growing technique is a depth-first search of an image, with the adjacency function being a color comparison based on finding the angle between vectors.

The color comparison operation treats an RGB triple as a vector, and when comparing it to another color, computes the angle between the two vectors. A small angle indicates similar colors. The magnitudes of these vectors are related to intensity. We compare the magnitudes of these vectors to determine if the intensities are close enough to call the colors similar. This comparison also makes the operation work well when the workspace is not lit uniformly.

4.1.3 Shadow Detection

We have developed a novel shadow detection technique in the context of this work, which is used to segment out shadow from an image. This segmentation aids in detection of food in the employee's hands.

Our shadow detection technique is useful when one expects that shadows may fall in a certain region of an image, and one knows a region that is touching these shadows. For instance, in this application, we know that the shadows of the arms and food will touch the workspace region. We know, based on the lighting of the scene, that there will be shadows cast by the arms and any food being carried, in many of the images in a sequence.

To operate, the technique requires a region from which to begin its search. This region is the one that is expected to touch the shadows. In this case, the potential region is the current workspace region. At the edge of this potential region, the change in intensity is computed. If the change in intensity is great enough (determined by applying a double-threshold), a depth-first search is initiated from this edge point. The adjacency function accepts changes in intensity that are small and positive, or non-positive. The result is that the technique detects intensity "valleys" that are touching the potential region.

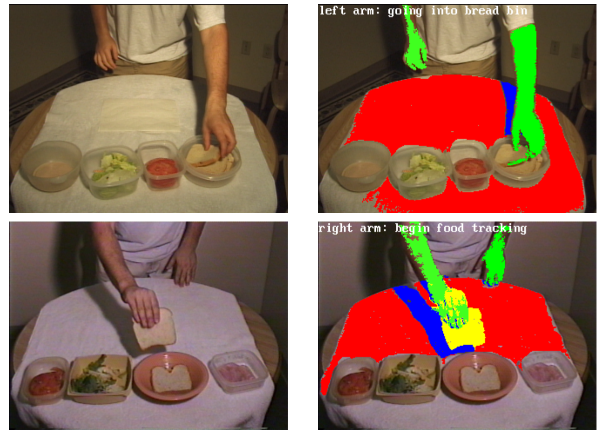

Figure 2: Sample shadow detection results; shadow regions shown in blue (from seq. 1 and 2).

The results of this simple operation can be improved by a simple technique. The mean intensity of all pixels detected as shadow is computed. As a post-processing step, all pixels previously marked as shadow that have an intensity greater than or equal to the mean are marked as not being shadow.

The shadow detection technique is utilized in two ways. First, it is used in segmentation, and second it is utilized in determining when an employee puts down a piece of food.

Please see sample results of the shadow detection operation in figure 2. This figure shows input images on the left, and output images on the right; shadow regions are marked in blue. The region marked in red is the potential region that was mentioned. In the first row, a small shadow region is cast by the left arm as it reaches into a food bin. The shadow detection procedure detects this region completely. The second row shows a sample frame from another sequence, this time under slightly different lighting conditions. This time, a large shadow region is cast by the arm and the food together. Once again, the shadow region that can be seen in the input image on the left is completely detected.

4.2. High-Level Module

The high-level part of the system accepts the results of the low-level vision techniques as input, and as output produces an interpretation of a video clip in terms of the construction of a sandwich.

The high-level module of the system must solve a number of problems. It must be able to determine when an employee is holding a piece of food and also when she is not.

It must be able to determine when and where a food item is placed on the workspace. When food items are stacked on top of each other, this arrangement should be recorded and reflected in the output.

In this section, we first present a simplified outline of the algorithm. Then, some important features of the algorithm are discussed in detail.

4.2.1 Algorithm Outline

Following is abbreviated pseudocode for the high-level algorithm, in outline form. This operation is carried out for each frame of a sequence.

Some of the operations mentioned in this outline require a detailed discussion. These discussions are provided in the following sections.

1. Obtain regions representing the arms, shadow, and workspace in the current frame (the basic features of each image in the sequence).
2. For each stack of food items the system has recorded as being placed on the workspace, find the region it currently occupies. This region is a "hole" in the workspace, or a region that is not detected as one of the three basic features. This "hole" must be contained entirely inside the true workspace, and will grow or shrink as a result of occlusion or other food items merging with it.
3. For each arm that the system has not marked as holding food:
   (a) If the hand did not leave one of the food bins this frame, then continue on with the next arm. However, if both arms have been processed, continue with the next frame of the sequence.
   (b) Otherwise, it is appropriate to search for food in the frame; apply the algorithm to detect food.
   (c) Screen the resulting food region candidates to determine if they represent food.
   (d) If food is detected, mark the arm as holding food.
4. For each arm that was holding food in the previous frame:
   (a) Search for food held in the hand by applying the algorithm to detect food.
   (b) If the region that was detected in previous frames has disappeared or if its area has dropped significantly:
      i. The food might have merged with one of the stacks of food on the workspace. Determine if this is the case, and if so, mark the arm as having merged food with that food stack.
   (c) If the arm is marked as having merged food with a food stack, determine if this is still the case by checking to see that the arm is still in contact with the food stack. Otherwise, remove the mark.
   (d) Using shadow and arm motion determine if the arm has put down the piece of food it was carrying. If so, keep a record of this.
      i. If the food was merged with a food stack, then add the food item that was being carried to the end of the list of food items in that stack.
      ii. Otherwise, create a new food stack, occupying the region of the food item.

4.2.2 Detecting Food Items in the Hand

Detecting that the employee has picked up food is an important part of the high-level module. The food detection technique must be able to find the region of a piece of food that is picked up, or determine that no food was picked up.

Food detection is applied at only a few stages in the algorithm. The idea behind this is to restrict searches to the most appropriate frames. These are frames when an arm has just left a food bin, or when an arm is marked as already carrying food.

The food detection procedure makes use of the output of all the low-level vision techniques. The first bit of information it makes use of is the *true workspace* region, W_t. Next is the workspace region detected in the current frame, W_c. The region consisting of both arm regions, A, and the union of all shadow regions S, are also needed. In addition, any regions that are currently occupied by food stacks (item 2 of the outline), the union of these represented by O, are not considered.

The union of possible food regions F is given by:

$$F = W_t \cap \overline{(W_c \cup A \cup S \cup O)} \tag{1}$$

where $\overline{B}$ indicates the complement of set B, and $\cup$, $\cap$ represent set union and intersection, respectively. It is also important to note that any candidate food region must be touching an arm.

This technique will pick up small error regions at the interface between skin, workspace, and shadow. These error regions are not recognized as skin, workspace or shadow because the colors blend (as a result of the quantization of a camera) and thus may not fit into any of the categories. To correct for this and other errors, each candidate food region is screened for validity. The first screening operation is to eliminate candidates that are extremely small.

Since the previously mentioned error regions occur at the interfaces between the skin and other regions, they are usually thin slivers. The second screening operation determines, for each pixel that is a food candidate, the smallest distance to a skin pixel. This is computed efficiently with a modified breadth-first search operation. If all candidate food pixels are located very close to the skin, the region must be an error.

The third screening operation eliminates food items that are not situated near the hand of an arm. If a food region touches an arm at the elbow, it will not be considered as a food candidate. To compute this predicate, we first find the vector V_a from the centroid of an arm to its lowest point. When arms are outstretched toward the food bins, this is a good approximation of the direction the arm is pointing. For each pixel in a candidate food region, we compute the angle between V_a and the vector between the centroid and the food pixel, V_f. By observing the distribution of angles over all candidate food pixels, we can decide if it is touching the arm in such a way that it might be grasped by the hand.

4.2.3 Determining When a Food Item is Released

After determining that a piece of food is picked up, the system must be able to determine when it is placed on the workspace. This information is used in determining how the food items are arranged on the workspace.

This description refers to step 4(d) in the outline; it deals with how the system determines that an employee has placed a piece of food on the workspace.

After having established that an employee is holding a piece of food, the easiest way to determine that the food has been put down is to wait for the arm to separate from that region in the 2-dimensional image. This simplistic approach by itself does not give satisfactory results in many sequences.

Consider the possibility that an employee places a food item on the workspace, but instead of moving her hand away from the food, reaches over it to pick up another piece of food. In this case, the hand will begin to occlude the food item that has been placed on the table, instead of parting with it. By the time the hand region is detected as parting with the food region (in this case because the hand has actually occluded the food), the system is not able to record the area that the food occupies in order to form a new food stack.

To handle this and other cases, the system uses a more sophisticated food tracking technique. Since the system can segment shadows reliably, the shadow region that an arm and the food it is holding casts is tracked. The area of the shadow region gives a measure of the height of the arm above the workspace. When the area is large, the arm must be well above the workspace. When the area is small, it must be covered by the arm, implying the arm is directly above or touching the workspace. The system also makes use of the motion of the arm (in the form of arm region change over time) to perform this tracking task. In the process of putting down a piece of food, the arm will first move to position the food, then pause while the food is placed on the workspace.

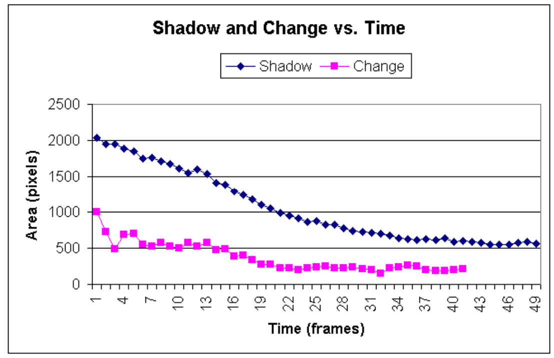

Figure 3: A plot of shadow and change area vs. time; data was collected while food was being carried to its destination on the workspace.

Figure 3 shows a plot of shadow and change area with respect to time. Shadow area is represented by the top line (marked with diamond shapes). This data was collected from a sequence while the employee was carrying a food item back to the workspace. The downward-sloping trend of this plot is a result of the arm moving closer to the workspace. The plot of change versus time is more erratic, but exhibits the same general behavior.

Another measure the system takes in order to fix the food occlusion problem is to record candidate regions that the food occupies in anticipation that the food will be released. Whenever shadow area reaches a new minimum during tracking (the arm has come closer to the workspace), the region that the food occupies is recorded. Then, if the arm occludes the food, this candidate region is used in preference of the current occluded food region.

5. Results

The accuracy of the system is measured by comparing the food item arrangement output to the actual sandwich that is created (as determined by the person playing the employee role). This output is a list of food stacks. Each food stack is a list of names of food items. This list of names gives, from bottom to top, the food items that make up a stack on the workspace. As mentioned above, the names of the food items come from the food bin labels supplied as configuration information. The system determines the name of a food item based on which bin it comes from.

Four sequences are discussed in this section. These sequences were created by the authors specifically for input to the system described here. All four follow the form discussed in 3.1. Several different food items are used through-

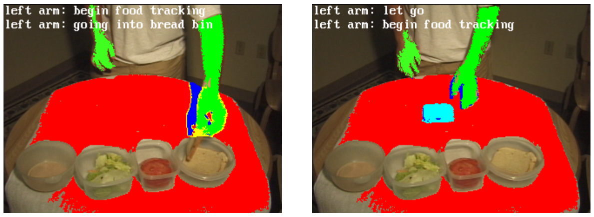

Figure 4: Sequence 1 highlights.

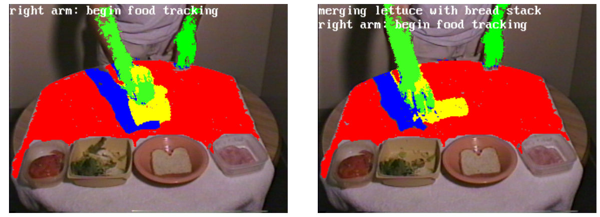

Figure 5: Sequence 2 highlights.

out the sequences.

All sequences are digitized at 320 by 240 resolution, 24-bit color, and 30 frames per second.

5.1. Sequence 1

The first sequence is of an employee making an opened-face sandwich that consists of a piece of bread with a piece of turkey on top. There are four food bins in this sequence; from left to right they are turkey, lettuce, tomato, and bread. These can be seen in figure 4.

At the beginning of the sequence, the employee reaches into the bread bin (on the far left) with his left hand, lifts out a piece of bread, and places it in the center of the workspace. The system correctly determines that the employee has picked up a piece of food, and tracks it until it is put down. Figure 4 shows frames at the beginning and end of the time interval that the bread is tracked.

Next, the employee places his right hand in the lettuce bin. He changes his mind, and retracts his hand without picking up any food. This event is correctly classified by the system. The piece of bread that was previously placed on the workspace is touching the arm, but is not detected as new food since it was previously tracked.

Finally, the employee reaches into the turkey bin with his right hand. He lifts out a piece of turkey, and places it on top of the piece of bread. The system correctly determines that the piece of turkey has merged with the piece of bread.

After completing the sequence, the system produces the following output describing the sequence: `[bread, turkey]`. This indicates the system detected that one stack of food items was arranged on the workspace, consisting of a piece of bread under a piece of turkey.

5.2 Sequence 2

In sequence 2, an employee creates a sandwich with lettuce, followed by a piece of ham, topped with tomato, and bread on top and bottom. Example images from the sequence can be seen in figure 5.

The sequence begins with the employee reaching into the ham bin. The employee removes his hand with no food, which the system correctly detects. The employee moves on to remove a piece of bread and place it on the workspace, then stack lettuce on top of it. The system follows these actions and records the changes to the workspace arrangement correctly. Next, the employee puts a piece of tomato on top of the sandwich. However, due to failure of the low-level vision techniques, the system fails to detect this. Despite the fact that the ham used in the sequence shares some color with the skin, the system is able to correctly detect that a piece is placed on top of the sandwich. The sequence concludes with the employee placing a piece of bread on top of the sandwich.

After completing processing of the sequence, the system produces the following output: `[bread, lettuce, ham, bread]`.

5.3 Sequence 3

Sequence 3 depicts an employee creating a sandwich with tomato, followed by lettuce, topped with salami, and bread on the top and bottom. Figure 6 gives example frames from this sequence.

The employee begins by placing his hand in the salami bin and then retracting it with no food, which is correctly interpreted by the system. Next, he takes a piece of bread from the bread bin and places it on the workspace (left-hand image, figure 6). Then, he places a piece of tomato on top of the bread. The tomato region is incorrectly detected as shadow, and so it fails to detect this event (right-hand image, figure 6). The employee moves on to put lettuce on top of the forming sandwich, which the system correctly interprets. The employee finishes by placing a piece of salami and finally a second piece of bread on the sandwich ; both events are correctly detected. The system produces `[bread, lettuce, salami, bread]` as its interpretation of the sequence.

5.4. Sequence 4

Sequence 4 depicts an employee building an open face sandwich with bread on bottom, then cheese, and topped with turkey. At the beginning of the sequence, the employee reaches into the bread and turkey bins without picking up food, and the system interprets these actions correctly. Then the employee places a piece of turkey on the workspace, stacks a piece of cheese on top of it, then finally

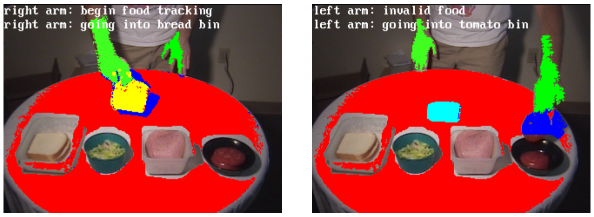

Figure 6: Sequence 3 highlights.

Figure 7: Sequence 4 highlights.

adds the turkey. The interpretation produced by the system is `[bread, cheese, turkey]`, which is correct.

6 Conclusions

We have presented a system to determine how a subject arranges objects on a workspace, and how this can be applied to understanding sequences of sandwich production. Some well known color vision techniques are employed, as well as a novel one for shadow detection. Using information gleaned from these operations, the system determines the actions of the human subject, as well as the arrangement of food items. We presented results from four sequences, which are reasonable and encouraging.

7 Future Work

There are many possibilities for extending the system. The system could be extended to have areas of interest in the image other than the food bins. Opposing arms, as well as food on the workspace, could serve the purposes of the food bins. In this way, the system could recognize when a food item moves from one hand to the other, or is picked up from the workspace and is moved around.

The authors feel that in addition to skin detection, some means of determining local motion might be employed to track the arms. Also, some sophisticated high-level processing coupled with edge detection might yield a method of segmenting arms from food with better results.

To improve the reliability of the high-level portion of the system, it could be extended to have some backtracking capabilities. Decisions could be delayed for several frames, with multiple possibilities considered in parallel. Then, at the end of the sequence, the system might determine what the best possibility is.

References

[1] Aggarwal, J. K. and Q. Cai, "Human Motion Analysis: A Review" *Computer Vision Image Understanding*, Vol.73, No.3, March, pp. 428-440, 1999.

[2] R. T. Collins, A. J. Lipton, and T. Kanade. Special section on video surveillance. *IEEE Transactions on PAMI*, 22(8):745–887, August 200.

[3] D. M Gavrila. "the visual analysis of human movement : A survey". *Computer Vision and Image Understanding*, 73:82–88, 1999.

[4] R. L. Rosin and T. Ellis, "Detecting and Classifying Intruders in Image Sequences," *Proc. British Machine Vision Conference*, pp. 24-26, 1991.

[5] Stephen S. Intille and Aaron F. Bobick, "Closed-World Tracking," *CVPR*, pp. 672-678, 1996.

[6] Stephen S. Intille, James W. Davis and Aaron F. Bobick, "Real Time Closed-World Tracking," *ICCV*, pp. 697-703, 1997.

[7] M. Izumi A. Kojiam. Generating natural language description of human behavior from video images. *Proceedings of International Conference on Pattern Recognition*, 4:728–731, 2000.

[8] Davis, J., and Shah, M. Visual gesture recognition. *IEE Proceedings Vision, Image and Signal Processing*, 141(2):101–106, 1994.

[9] Aaron Bobick and James W. Davis. Action recognition using temporal templates, pages 125–146. CVPR-97, 1997.

[10] Rick Kjeldsen and John Kender, "Finding Skin in Color Images," *Int. Workshop on Face and Gesture Recognition*, pp. 144-150, 1996.