J. Intell. Syst. **20** (2011), 79–100
DOI 10.1515/JISYS.2011.006 © de Gruyter 2011

# A Novel Efficient Algorithm for Locating and Tracking Object Parts in Low Resolution Videos

David O. Johnson and Arvin Agah

**Abstract.** In this paper, a novel efficient algorithm is presented for locating and tracking object parts in low resolution videos using Lowe's SIFT keypoints with a nearest neighbor object detection approach. Our interest lies in using this information as one step in the process of automatically programming service, household, or personal robots to perform the skills that are being taught in easily obtainable instructional videos. In the reported experiments, the system looked for 14 parts of inanimate and animate objects in 40 natural outdoor scenes. The scenes were frames from a low-resolution instructional video on cleaning golf clubs containing 2,405 frames of 180 by 240 pixels. The system was trained using 39 frames that were half-way between the test frames. Despite the low resolution quality of the instructional video and occluded training samples, the system achieved a recall of 49 % with a precision of 71 % and an F1 of 0.58, which is better than that achieved by less demanding applications. In order to verify that the reported results were not dependent on the specific video, the proposed technique was applied to another video and the results are reported.

**Keywords.** Object detection, object recognition, object tracking, scale-invariant feature transform (SIFT), nearest neighbor algorithm.

**2010 Mathematics Subject Classification.** 65D19.

## 1 Introduction

Locating and tracking human body parts (e.g., hand, head, and torso) and inanimate objects across the frames of a video has applications in areas such as video surveillance and multimedia content analysis and retrieval. Our interest lies in using this information as one step in the process of automatically programming service/household/personal robots to perform the skills that are being taught in easily obtainable instructional videos.

Much work has been done in recognizing stationary objects against stationary backgrounds. In this work, still photographs of objects are used to train the system, and then the system is used to recognize the objects in still photographs of scenes. The scenes usually contain multiple objects, some of which the system is trained to recognize. Usually, some of the objects are occluded. Research work has been done with surveillance videos in recognizing moving objects (e.g., cars, people)

against stationary backgrounds. The challenging subject of this paper is recognizing moving objects against moving backgrounds. This type of situation usually occurs in videos taken with a moving camera. Examples of these types of videos are entertainment videos, instructional videos, and surveillance videos where the camera is moving. Analysis of these types of videos requires addressing issues with blurring caused by camera movement and video compression techniques. In addition, camera zooming and panning cause the size and view angle of images to change rapidly. Except for the entertainment videos, these types of videos are generally of low resolution, which makes object detection even more difficult. We are interested in tracking not only the whole objects, but parts of inanimate objects (e.g., brush handle and brush bristles) and parts of the human body. This is important because, for example, in order to learn how to clean a table top with a brush, the robot must be able to recognize that the human in the instructional video has grabbed the brush by the handle with his or her hand and then used the brush to clean the table top. Schügerl et al. referred to this as object re-detection, which aims at finding occurrences of specific objects in a single video [15]. In contrast to general object identification, the samples for learning an object are taken directly from the video on which the object re-detection is performed. Because of this, many of the training samples are occluded, which complicates the training of the system.

Our algorithm locates and tracks object parts in low resolution videos in a number of steps. First the object parts we want to locate are learned by extracting local feature descriptors from images of these objects and storing them in an object database. We used SIFT keypoints as defined by Lowe [9] for the descriptors. To recognize object parts in a video frame, SIFT keypoints are identified in the video frame and matched against the SIFT keypoints in the object database. The best matching descriptor pairs are identified using a nearest neighbor approach which is explained in detail later. In order to minimize the false-positives, we also use a threshold to trim the list of potential matches between the keypoints in the image and those in the object database.

In experiments reported in this paper using this algorithm, the system looked for 14 parts of inanimate and animate objects in 40 natural outdoor scenes. The scenes were frames from a low-resolution instructional video containing 2,405 frames of 180 by 240 pixels. The system was trained using 39 frames that were half-way between the test frames (e.g., frames 30 and 90 were test frames and frame 60 was a training frame). Despite the poor quality of the instructional video and occluded training samples, the system achieved a recall of 49 % with a precision of 71 % and an F1 of 0.58. F1 is the harmonic mean of precision and recall:

$$\mathrm{F1} = 2 \cdot \mathrm{precision} \cdot \mathrm{recall}/(\mathrm{precision} + \mathrm{recall}). \quad (1)$$

In order to verify that the reported results were not dependent on the specific video, the proposed technique was applied to another video and the results are reported.

The current state of art in object recognition uses more sophisticated techniques than SIFT features alone, and achieves much higher rates of object detection. However, since in our research this is only one step in the process of automatically programming robots to perform the skills that are being taught in instructional videos, we were looking for a simple, but effective technique. Our results show that using only SIFT features we can achieve object detection rates that are better than those achieved in less demanding applications where the video resolution is higher, fewer objects are detected in each frame, and whole objects instead of parts of objects are detected.

This paper is organized into six sections. Section 2 discusses the related work. Section 3 presents our research approach, describing the object recognition algorithms that we employed, the experimental setup, building the required database, and the implementation of our object recognition algorithms. Section 4 discusses the results of the experiments. Section 5 presents a collection of other methods and procedures that were tried, but were not successful. Section 6 concludes the paper, with a comparison of the results that were obtained with those of other researchers, and includes a brief discussion of the future work.

## 2 Related Works

It was stated in the Interactive Robot Learning Workshop, The Robotics: Science and Systems 2008, held in Zurich, Switzerland in June, 2008 that "Many future applications for autonomous robots bring them into human environments as helpful assistants to untrained users in homes, offices, hospitals, and more. These applications will often require robots to flexibly adapt to the dynamic needs of human users. Rather than being pre-programmed at the factory with a fixed repertoire of skills, these personal robots will need to be able to quickly learn how to perform new tasks and skills from natural human instruction. Moreover, it is our belief that people should not have to learn a new form of interaction in order to teach these machines, that the robots should be able to take advantage of communication channels that are natural and intuitive for the human partner." We are in full agreement with that statement. One method by which robots learn from natural human instruction is called Robot Programming by Demonstration (RbD), also referred to as Learning by Imitation and Programming by Demonstration (PbD). RbD is a non-verbal technique, generally used to teach new motor skills to a robot. Billard et al. produced an extensive review of RbD [1], including a history of RbD, the current approaches to RbD, and the open issues. Inamura et al. provide an example of

RbD with an autonomous robot, using Hidden Markov Models to encode human joint trajectories into proto-symbols [6]. Each proto-symbol represented one type of motion (e.g., squatting, walking, picking something up, Cossack dancing). In their experiments, they were able to show that a HRP-2W humanoid robot could learn and demonstrate the proto-symbols by watching a human perform it [6]. In their work, the joints of the human teacher were tracked using a wearable motion capturing system, which usually does not happen in instructional videos. The goal of the research reported in this paper is to use Lowe's SIFT keypoints to identify and track the trajectories of objects (e.g., scrub brush, pail of water) and major body parts (e.g., hand, head, body) in instructional videos [9]. Then we can use a Machine Learning technique (e.g., Hidden Markov Models) to recognize the trajectories as proto-symbols.

There are two main approaches to recognizing objects, namely, model-based and appearance-based [15]. Model-based approaches use three-dimensional models to represent an object with geometric features such as lines, vertices and ellipses. Appearance-based approaches use local feature descriptors–either texture or color–to represent the object. Appearance-based approaches are used because they are insensitive to small changes in rotation, scale, and lighting; and they work well when the objects are partially occluded. Lowe's Scale-Invariant Feature Transform (SIFT) is one of the more widely used texture descriptors [9]. Jensfelt's et al. work is a typical example of using a color descriptor [7]. A typical object recognition system that works with local feature descriptors performs the recognition task in a number of steps [15]. First some objects of interest are learned by extracting local feature descriptors from images of these objects and storing them in an object database. To recognize objects in a test image, local feature descriptors are extracted from the test image and matched against the descriptors in the object database. After the best matching descriptor pairs are identified, an optional verification step can be performed to decide whether or not an object appears in the test image. The algorithm discussed in this paper uses Lowe's SIFT texture descriptors.

As an example of using SIFT to classify images, van de Weijer and Schmid conducted experiments on two data sets [19]: a bird data set containing six classes of bird species, with 100 images per species, divided into 300 training and 300 testing images; and a soccer team data set containing images from seven soccer teams, with 40 images per team, divided into 25 training and 15 testing images. Both data sets consisted of low-quality images obtained from the Internet. They reported recalls of 43 % and 55 % on the soccer teams and the birds, respectively. The significant differences between their work and ours are: (1) we identified multiple objects in a scene, whereas they only identified one (birds or soccer teams); (2) we identified parts of whole objects (both people and inanimate objects), whereas they

identified whole objects; and (3) they classified an image as belonging to 1 of $n$ classes, whereas we located $m$ of $n$ objects in an image.

Much research has been done on recognizing moving objects against stationary backgrounds, especially in the area of video surveillance. Hu et al. surveyed 185 articles and concluded that the processing framework of visual surveillance in dynamic scenes includes the following stages [4]: modeling of environments, detection of motion, classification of moving objects, tracking, understanding and description of behaviors, human identification, and fusion of data from multiple cameras. They reviewed recent developments and general strategies of all these stages. The major differences between this type of work and our approach are that (1) the background does not usually move in surveillance videos; and (2) surveillance applications are trying to identify larger moving objects (e.g., cars or people), whereas, we are trying to identify parts of objects (e.g., brush handle, brush bristles, body parts).

Closer to our approach, Schügerl et al. proposed an object re-detection approach using SIFT and MPEG-7 color descriptors that were extracted around the same interest points [15]. They evaluated the approach on two different data sets and showed that the MPEG-7 ColorLayout descriptor performs best of the tested color descriptors and that the joint approach yields better results than the use of SIFT or color descriptors only. Although Schügerl et al. reported better results using a combination of SIFT and color descriptors, they noted that most local feature approaches are only based on textural information, because color descriptors are not invariant to different lighting conditions such as shadows or illumination. Due to the fact that the lighting conditions varied significantly in our application, we chose to use only SIFT descriptors. In their experiments with high resolution images, they reported a precision of 75 % and a recall of 46 % on a data set with one training sample of a car model and 50 test images using SIFT descriptors only. They reported a precision of 70 % and a recall of 32 % on a data set of people using only SIFT descriptors. The test data set contained 742 video scenes of people walking inside a building and random shots taken from the TRECVID test data set, which includes different people in various scenes [17]. The significant differences between their work and ours are: (1) we identified multiple objects in a scene, but they only identified one (cars or people); (2) we used low resolution video, while they used high resolution; and (3) we identified parts of whole objects (both people and inanimate objects), whereas they identified whole objects.

Savarese and Fei-Fei proposed a method for identifying parts of objects [13], linking together parts of objects using their mutual homographic transformation. Each part was defined by a collection of SIFT descriptors that was discovered during the training process. The training process included analyzing the object from various angles and using homographic transforms to discover the salient parts.

They used 48 training images per object (eight viewing angles, three heights and three scales). In an eight category classification task where they randomly selected seven object instances ($\sim$ 280 images) to build the model, and four object instances ($\sim$ 70 images) for testing, they reported precision values ranging between 62 % and 81 %, with an average of 75.65 %. The significant differences between their work and ours are: (1) we identified parts of whole objects, but they used parts to identify whole objects; (2) we identified moving parts against a moving background, while they identified stationary parts against a stationary background; (3) we identified parts of human bodies and inanimate objects, but they only recognized parts of inanimate objects; (4) we identified multiple parts of multiple objects in each test scene, whereas they recognized parts of a single object. Savarese and Fei-Fei showed that their method outperformed the "bag of words model" used by Thomas et al. [18]. Thomas et al. combined the Implicit Shape Model for object class detection proposed by Leibe and Schiele [8] with the multi-view specific object recognition system of Ferrari et al. [3].

Zickler and Efros used SIFT in conjunction with principal component analysis (PCA) and a clustered voting scheme to detect a deformable object (robot) in medium resolution videos (320-by-240 pixels captured at 30 frames per second) [21]. They reported recall values ranging between 5 % and 100 %. The significant differences between their work and ours are: (1) we identified multiple objects in a scene, but they only identified one; (2) we identified parts of whole objects (both people and inanimate objects), whereas they identified whole objects; (3) we used low resolution video, while they used a higher resolution. They also noted that they achieved high recall values because there was only a small inter-class variation in terms of texture. This was because in each case they were only trying to detect instances of one model of a robot which, except for the joint configurations, were identical in each frame of the video. This is not true with human body parts, where a high degree of inter-class variation of shape, texture, and color exists. Although they did not analyze the performance of their algorithm for these types of classes, they hypothesized that their approach would lose some of its effectiveness when trained on only a small subset of instances of human body parts, as we did in our experiments.

## 3 Research Approach

### 3.1 Object Recognition Algorithms

We used SIFT keypoints as defined by Lowe [9] for the descriptors. SIFT keypoints consist of the location ($x$ and $y$ coordinate) of the keypoint in the training image, the scale of the keypoint, its orientation in radians, and a 128 dimension de-

| Part/View | Distance | x | y |
|---|---|---|---|
| torso1 | 504 | 253 | 150 |
| right upper arm41 | 945 | 118 | 90 |
| pail of water66 | 1002 | 135 | 132 |
| torso1 | 1326 | 69 | 47 |
| pail of water66 | 1357 | 126 | 153 |
| torso1 | 1975 | 164 | 99 |
| right upper arm41 | 1992 | 81 | 72 |
| head41 | 2127 | 162 | 93 |
| torso1 | 2216 | 235 | 31 |
| pail of water66 | 2318 | 227 | 150 |
| head3 | 2946 | 51 | 121 |
| torso1 | 3457 | 152 | 52 |
| torso1 | 3586 | 24 | 124 |
| pail of water66 | 3897 | 271 | 126 |
| torso1 | 3924 | 167 | 97 |
| torso40 | 3979 | 256 | 152 |
| torso3 | 4210 | 99 | 143 |

Table 1. Example of nearest neighbor object recognition algorithm.

scriptor. The 128 dimension descriptor describes the gradient orientation of the 16 regions (4 by 4) centered on the location of the keypoint. Two or more keypoints can have the same location, but must be different in terms of scale, orientation, or descriptor. Each object part (e.g., brush bristle, brush handle, head, torso) in the Parts Database – list of known parts – is represented as a set of SIFT keypoints from a single training image. Multiple views of the same part are gathered from additional training images. Thus, the Parts Database consists of many sets of SIFT keypoints, with each set representing a single view of an object part.

We used a nearest neighbor approach to detect parts in an image [16]. First, the Euclidean distance, or L2 norm, between each SIFT keypoint in the test image and each SIFT keypoint in the Parts Database is calculated. The part view with the shortest distance is assumed to be the view of the part detected in the test image. The centroid of the part in the image is then calculated by taking the mean of the other keypoints in the image that are matched to other keypoints of the assumed part view. All other part views are ignored. Therefore, this algorithm finds at most one instance of an object part in an image. For example, the distances have been calculated as shown in Table 1, where view number 1 for the body part torso is represented as torso1.

Since the keypoint at the $x$-$y$ coordinates of (253, 150) has the shortest distance to its matching keypoint in the Parts Database, torso1 is the view of the torso recognized. The centroid is then calculated, using the $x$-$y$ coordinates of the other torso1 keypoints detected, to be (152, 85.7). The keypoints for torso40 and torso3 are ignored.

In order to minimize the false-positives, we also use a threshold called MatchThresh, ranging from 0 to 100 %, to trim the list of potential matches between the keypoints in the image and those in the Parts Database. Only the shortest MatchThresh percent are considered in the algorithm. For example, if MatchThresh is 75 %, then only the shortest 13 in Table 1 would be considered, i.e., pail of water66, torso1, torso40, and torso3 at the bottom of the list would be ignored. In this example, MatchThresh would only affect the calculation of the centroid for torso and pail of water. However, in other cases the first view of a part might fall above the threshold, and therefore the part would not be detected at all. Using a percentage of the shortest distances as a threshold produced better results instead of the maximum distance. There are a number of techniques for minimizing false matches for SIFT. Some of these rely on information from multiple frames. Our method uses only information from the frame under consideration. An area for further study in our research is to examine techniques that use information from multiple frames and use more sophisticated statistical methods.

As discussed earlier, this work is part of a larger project to demonstrate a humanoid robot learning and demonstrating types of motion by watching a human perform them in an instructional video. Inamura et al. accomplished this with the joints of the human teacher being tracked using a wearable motion capturing system, which usually does not happen in instructional videos [5]. Thus, we must track the motion of the human body parts and inanimate objects using the points extracted from the video itself. The exact definition of the point on the wearable motion capturing system (i.e., is it the geometric center of the elbow or is it the center of mass of the elbow) is not important as long as it is in the general vicinity of the joint being tracked and consistent over time. Likewise, where the point we are tracking is located on the human body part or inanimate object is not important as long as it is in the general vicinity and is consistent over time. The definition of the centroid, or point, we track is closer to the definition used in physics, rather than that used in geometry. In physics, the centroid is the center of mass, which is the average of all points weighted by the local density. In our case, the density is represented by the number of keypoints detected at a specific location in the image.

If all of the keypoints detected in the training image of the part are present in the test image, then the centroid of the test image of the part will be consistent with the centroid of the training image of the part. However, due to occlusion,

rotation, or noise in the test image, not all keypoints of a part may be detected, and the test image centroid will not be consistent with the training image centroid. Intuitively, to account for undetected keypoints, it would seem better to calculate the centroid using spatial transforms rather than using the mean. Initially, we used spatial transforms to calculate the centroid when not all the training keypoints were detected, i.e., linear conformal transform when two non-collocated keypoints were detected, affine transform when three non-collocated keypoints were detected, and projective transform when four or more non-collocated keypoints were detected. However, the detection results were slightly worse and considerably more computationally intensive rather than using the simpler calculation of the mean of the detected keypoints.

### 3.2 Experimental Setup

For the experiments, we used frames from an instructional video on how to clean a golf club. The instructional video was downloaded from `eHow.com` [2] using `savevid.com` [14]. `eHow.com` is a Website which provides the ability to research, share, and discuss instructional solutions that help complete day-to-day tasks and projects. `savevid.com` is a tool which enables downloading videos from streaming video sites. We converted the video from .flv format to .avi format using the AVS Video Converter from Online Media Technologies, Ltd. Online Media Technologies Ltd. is a developer of software for digital video and multimedia processing [11]. The resulting video contained 2,405 frames, recorded at 29.97 frames per second. Each frame was 240 pixels wide and 180 pixels high.

The 14 object parts used in the experiments were selected from frame 1000, which was arbitrarily selected from the middle of the video. Figure 1 shows the object parts identified on frame 1000.

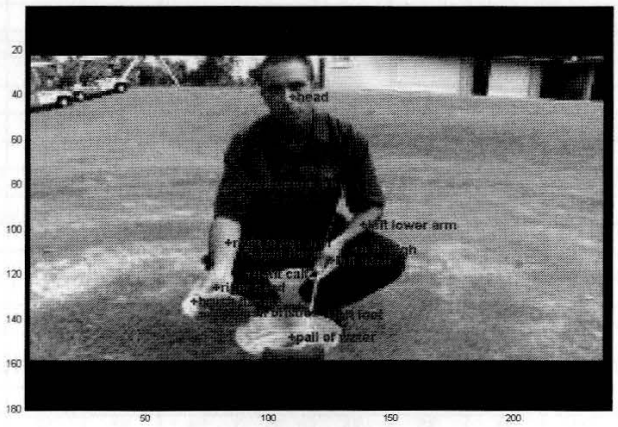


Figure 1. Frame 1000 showing object parts used in the experiments.

In order to be more realistic, the object parts were selected to include objects with parts (brush handle and brush bristles) and objects without parts (pail of water), body parts without the matching symmetrical part (right upper arm without the matching left upper arm), body parts with the matching symmetrical part (right thigh and left thigh), body parts relevant to the task (right-hand and left-hand), body parts not relevant to the task (head and left foot), and inanimate objects relevant to the task (brush and pail of water).

Every sixtieth frame, starting with frame 30, was used to form the 40 test frames. Table 2 shows the ground truth of the parts that are visible in each frame.

The test frames included a variety of perspectives for each part, as illustrated in Figure 2.

| Frame | brush bristles | brush handle | head | left foot | left-hand | left lower arm | left thigh | pail of water | right calf | right-hand | right lower arm | right thigh | right upper arm | torso | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 30 | | | 1 | | | 1 | | | | | | | 1 | 1 | 4 |
| 90 | | | 1 | | | 1 | | | | | 1 | | 1 | 1 | 5 |
| 150 | | | 1 | | | 1 | | | | | 1 | | 1 | 1 | 5 |
| 210 | | | 1 | | | 1 | | | | | 1 | | 1 | 1 | 5 |
| 270 | | | 1 | | | 1 | | | | | 1 | | 1 | 1 | 5 |
| 330 | | | 1 | | | 1 | | | | | 1 | | 1 | 1 | 5 |
| 390 | | | 1 | | 1 | 1 | | | | | 1 | | 1 | 1 | 6 |
| 450 | | | 1 | | 1 | 1 | | | | 1 | 1 | | 1 | 1 | 7 |
| 510 | | | 1 | | 1 | 1 | 1 | | | 1 | 1 | 1 | 1 | 1 | 9 |
| 570 | | | | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | | 1 | 8 |
| 630 | | | | | 1 | 1 | 1 | | | 1 | 1 | 1 | 1 | 1 | 8 |
| 690 | | | 1 | | 1 | 1 | 1 | | | 1 | 1 | 1 | 1 | 1 | 9 |
| 750 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 1 | 12 |
| 810 | 1 | 1 | | | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | | | 9 |
| 870 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 11 |
| 930 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 990 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 1050 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 1110 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 11 |
| 1170 | 1 | 1 | | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | | | 10 |
| 1230 | | 1 | | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | | | 9 |
| 1290 | 1 | 1 | | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | | | 10 |
| 1350 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 11 |
| 1410 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 1 | | | 10 |
| 1470 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 1530 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 11 |
| 1590 | 1 | 1 | | 1 | 1 | | | 1 | 1 | | | | | | 6 |
| 1650 | | | 1 | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 10 |
| 1710 | | | 1 | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 10 |
| 1770 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 1830 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 11 |
| 1890 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 11 |
| 1950 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | 11 |
| 2010 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 2070 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 2130 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 2190 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 2250 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 2310 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 2370 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| Total | 25 | 26 | 21 | 25 | 34 | 35 | 31 | 26 | 29 | 32 | 37 | 31 | 25 | 27 | 404 |

Table 2. Ground truth of the parts that are visible in each frame.

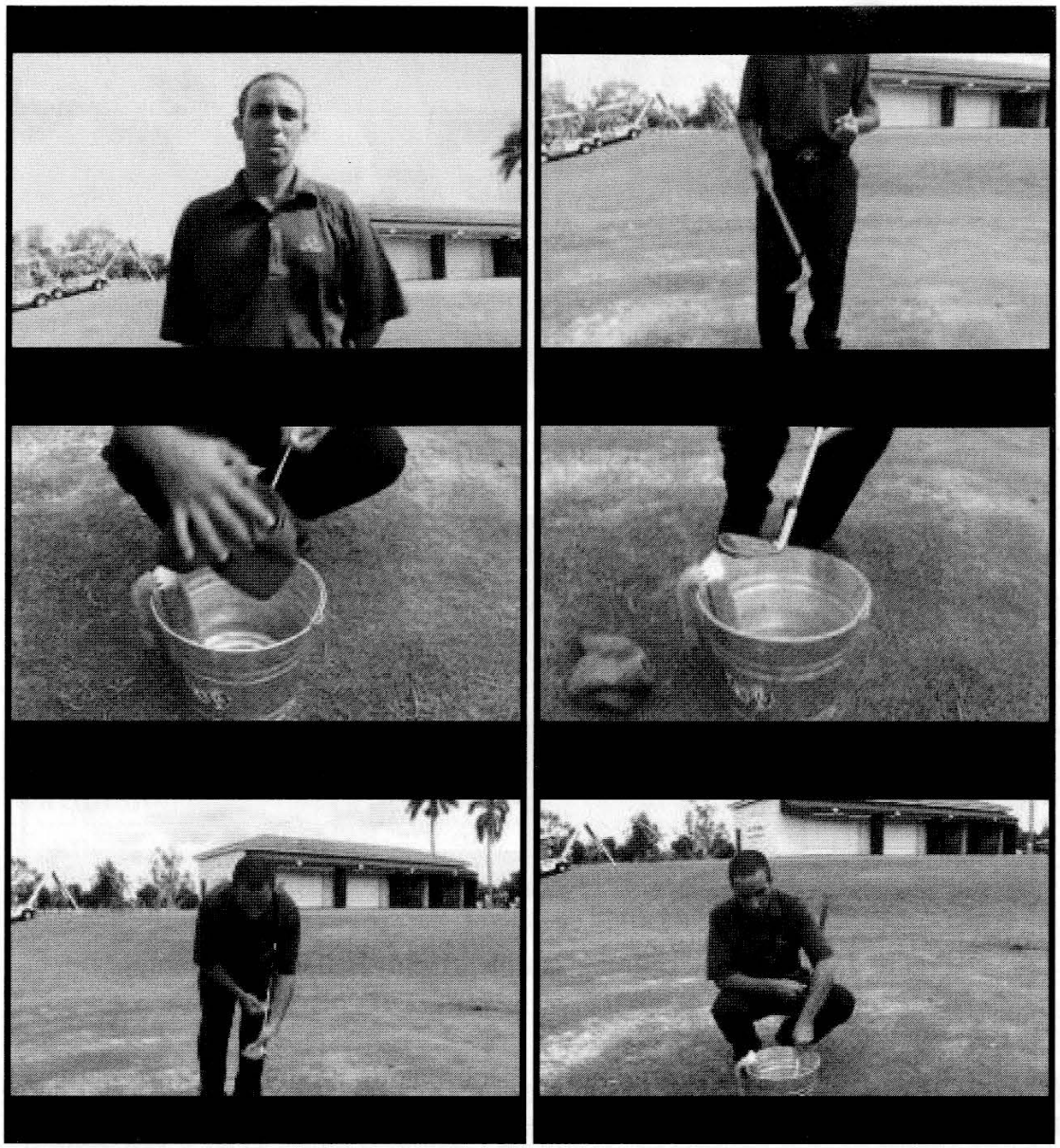

Figure 2. Examples of test images showing different perspectives.

### 3.3 Building the SIFT Keypoint Database (Parts Database)

Every sixtieth frame, starting with frame 60, was used for the 39 training frames. Thus, each training image is exactly between two test frames, making object detection as difficult as possible. The Parts Database was built by first calculating the SIFT keypoints for each training image, using the vl_sift tool from VLFeat.org [20]. For vl_sift, we set the edge threshold to 20 and left the peak threshold at its default value of 0. The edge threshold eliminates peaks of the DoG (Difference of Gaussians) scale space whose curvature is too small, since such peaks yield badly localized frames. Increasing edge threshold increases the number of keypoints detected. The peak threshold filters peaks of the DoG scale

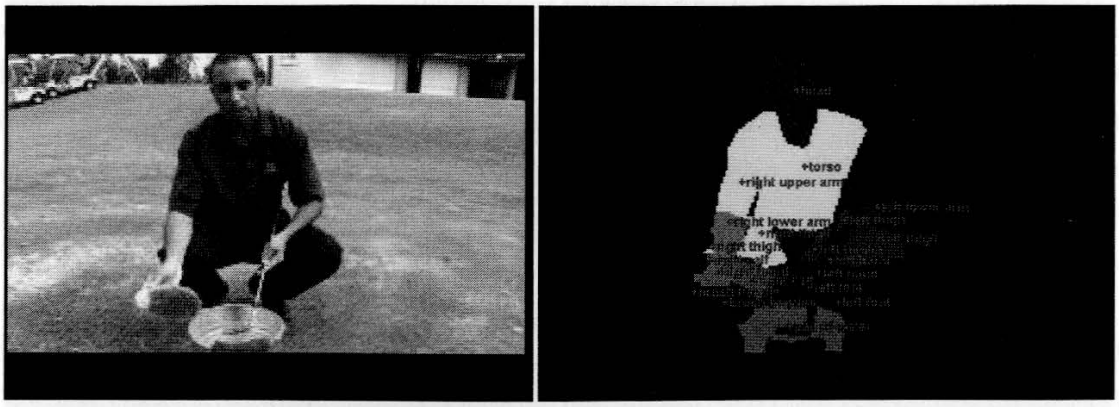


Figure 3. (left) Training image 1005, (right) Region of interest for each part.

Figure 4. (left) Good training sample, (right) Bad training sample.

space that are too small in absolute value. Decreasing peak threshold increases the number of keypoints detected.

The SIFT keypoints from each training image were grouped into parts using polygons to identify regions of the image that belonged to each part. We used the MATLAB roipoly function [10], but any method including manual assignment could be used to group them. The MATLAB roipoly function allows the user to manually identify an arbitrary number of points on the boundary of an object with a cursor and then MATLAB constructs a polygon by drawing straight lines between adjacent points. Figure 3 shows an example of the polygons used to group the keypoints in training frame 1005 into parts. Multiple polygons were used for a part that was occluded as illustrated by the left foot in Figure 3.

The parts from each training image represented one view of the part. By arbitrarily using the training frames exactly between two test frames, we ended up with some good training images and some bad ones, as illustrated in Figure 4.

As shown in Table 3, the resulting Parts Database contained from 12 to 32 views per part; and each view of a part contained from 1 to 30 keypoints.

| | | | Keypoints/View | | |
|---|---|---|---|---|---|
| **Part Name** | **Views** | **Keypoints** | **Minimum** | **Maximum** | **Mean** |
| brush bristles | 19 | 50 | 1 | 6 | 2.6 |
| brush handle | 12 | 25 | 1 | 4 | 2.1 |
| head | 20 | 78 | 1 | 8 | 3.9 |
| left foot | 17 | 33 | 1 | 3 | 1.9 |
| left-hand | 30 | 85 | 1 | 7 | 2.8 |
| left lower arm | 26 | 48 | 1 | 5 | 1.8 |
| left thigh | 29 | 112 | 1 | 12 | 3.9 |
| pail of water | 26 | 225 | 1 | 20 | 8.7 |
| right calf | 25 | 74 | 1 | 9 | 3.0 |
| right-hand | 29 | 90 | 1 | 9 | 3.1 |
| right lower arm | 32 | 46 | 1 | 4 | 1.4 |
| right thigh | 30 | 85 | 1 | 6 | 2.8 |
| right upper arm | 23 | 49 | 1 | 5 | 2.1 |
| torso | 27 | 340 | 1 | 30 | 12.6 |
| **Minimum** | 12 | 25 | 1 | 3 | 1.4 |
| **Maximum** | 32 | 340 | 1 | 30 | 12.6 |
| **Mean** | 25 | 96 | 1 | 9 | 3.8 |

Table 3. SIFT keypoint database (Parts Database) statistics.

### 3.4 Locating Object Parts

The object recognition algorithm was implemented using the VLFeat vl_sift tool to calculate SIFT keypoints, and the VLFeat vl_ubcmatch tool to match keypoints in the test image with keypoints in the Parts Database [20]. For vl_sift, we set the edge threshold to 20 and left the peak threshold at its default of 0. We used the default values for vl_ubcmatch. The remainder of the algorithm was implemented using standard MATLAB functions [10]. The results were 40 lists of detected parts in each test frame, along with the corresponding $x$-$y$ coordinates of the centroid.

### 3.5 Validating the Approach with Another Video

In order to illustrate that the obtained results were not dependent on the video, we applied the technique to a different video of another person demonstrating an additional method of cleaning a golf club. The second instructional video was downloaded from `eHow.com` [2]. We converted the video from .flv format to .avi format using the AVS Video Converter from Online Media Technologies, Ltd [11]. The resulting video contained 1,964 frames, recorded at 29.97 frames per second.

Each frame was 240 pixels wide and 180 pixels high. We used the same 11 body parts as in the first video, but with three different inanimate objects due to the different methods used to clean a golf club. Specifically in the second video the inanimate objects were: a small brush, golf club, and towel, whereas in the first video we used: brush bristles, brush handle, and pail of water. In the second video, we did not break the brush into two parts (bristles and handle) because the brush was too small to discern them in the video. Table 4 shows the ground truth of the parts that are visible in each frame of the second video.

| Frame | brush | golf club | head | left foot | left-hand | left lower arm | left thigh | right calf | right-hand | right lower arm | right thigh | right upper arm | torso | towel | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 24 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 73 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 122 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 171 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 220 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 269 | 1 | 1 | | | 1 | 1 | | | 1 | 1 | | 1 | 1 | | 8 |
| 318 | 1 | 1 | | | 1 | 1 | | | | 1 | | 1 | 1 | | 7 |
| 367 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 416 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 465 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 514 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 563 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 612 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 661 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 710 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 759 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 808 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 857 | 1 | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | | 10 |
| 906 | 1 | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | | 10 |
| 955 | 1 | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | | 10 |
| 1004 | 1 | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | | 10 |
| 1053 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 1102 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 1151 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 1200 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 14 |
| 1249 | | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 10 |
| 1298 | | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | | 1 | 1 | 9 |
| 1347 | | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | | 1 | 1 | 9 |
| 1396 | | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | | 1 | 1 | 9 |
| 1445 | | 1 | | | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 | 10 |
| 1494 | | 1 | | | 1 | 1 | 1 | | 1 | 1 | | 1 | 1 | | 8 |
| 1543 | | 1 | | | 1 | 1 | 1 | | 1 | 1 | | 1 | 1 | | 8 |
| 1592 | | 1 | 1 | | 1 | 1 | | | 1 | 1 | | 1 | 1 | | 8 |
| 1641 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 1690 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 1739 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 1788 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 1837 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 1886 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| 1935 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 13 |
| Total | 22 | 40 | 27 | 26 | 40 | 40 | 37 | 26 | 39 | 40 | 35 | 37 | 40 | 31 | 480 |

Table 4. Ground truth of the parts that are visible in each frame of the second video.

## 4 Experimental Results

Typically in object recognition experiments, bounding boxes are used to determine whether a detected object is a true-positive or a false-positive. Because of the odd shapes of the parts and our goal of increased accuracy, we used bounding polygons to determine true-positives from false-positives. For each test frame, we used the MATLAB roipoly tool [10] to define the bounding polygons for each part, as illustrated in Figure 5.

If the centroid fell within the boundaries of the bounding polygon, then it was considered a true-positive, otherwise it was considered a false-positive.

We used the standard metrics of precision (P), recall (R), and F1 formulas for our measurements:

$$\mathrm{TP} = \text{number of true-positives} \tag{2}$$

$$\mathrm{FP} = \text{number of false-positives} \tag{3}$$

$$\begin{aligned}\mathrm{FN} &= \text{number of false-negatives} = \text{Ground Truth (from Table 2)} - \mathrm{TP} \\ &= 404 - \mathrm{TP}\end{aligned} \tag{4}$$

$$\mathrm{P} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP}) \tag{5}$$

$$\mathrm{R} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN}) \tag{6}$$

$$\mathrm{F1} = 2\mathrm{PR}/(\mathrm{P} + \mathrm{R}) \tag{7}$$

Table 5 shows the experimental results for various values of MatchThresh.

Figure 6 illustrates the same results as a graph.

Figure 5. Bounding polygon (marked) for pail of water in test image 1050.

| MatchThresh | 10 % | 20 % | 30 % | 40 % | 50 % | 60 % | 70 % | 80 % | 90 % | 100 % |
|---|---|---|---|---|---|---|---|---|---|---|
| Recall | 16 % | 21 % | 26 % | 28 % | 33 % | 39 % | 44 % | 46 % | 49 % | 49 % |
| Precision | 90 % | 87 % | 82 % | 80 % | 75 % | 75 % | 75 % | 74 % | 71 % | 69 % |
| F1 | 0.27 | 0.34 | 0.4 | 0.42 | 0.46 | 0.51 | 0.56 | 0.57 | 0.58 | 0.57 |

Table 5. Experimental Results for various values of MatchThresh.

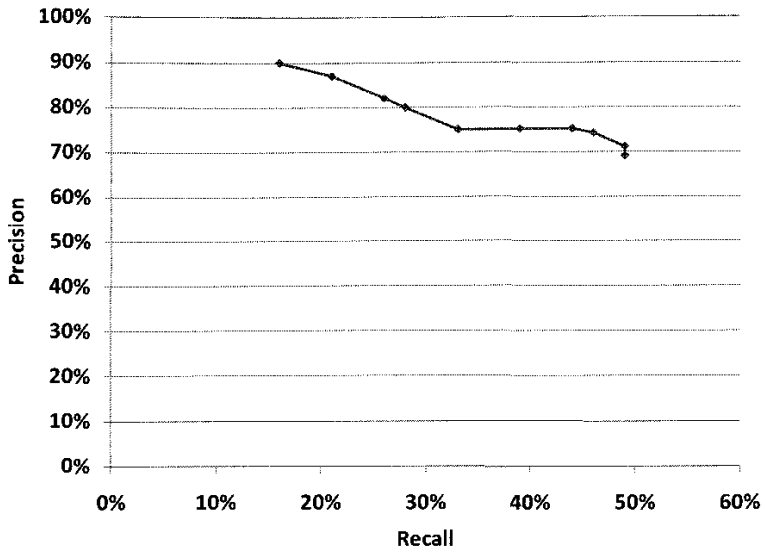


Figure 6. Experimental Results for various values of MatchThresh.

For the second video, we tested it with the optimum MatchThresh for the first video of 90 %. The results were a recall of 42 %, a precision of 72 %, and an F1 of 0.53.

Table 6 shows that these results are better than those achieved in less demanding applications.

The application used by Schügerl et al. is less demanding in that [15]: (1) we identified multiple objects in a scene, but they only identified one (cars or people); (2) we used low resolution video, while they used high resolution; and (3) we identified parts of whole objects (both people and inanimate objects), whereas they identified whole objects. The application used by Weijer and Schmid is less demanding in that [19]: (1) we identified multiple objects in a scene, whereas they only identified one (birds or soccer teams); (2) we identified parts of whole objects (both people and inanimate objects), whereas they identified whole objects; and (3) they classified an image as belonging to 1 of $n$ classes, whereas we located $m$ of $n$ objects in an image.

| Experiments | Recall | Precision | F1 | Data Set | Objects per Scene | Resolution | Parts/ Whole |
|---|---|---|---|---|---|---|---|
| Ours (video 1) | 49 % | 71 % | 0.58 | body parts, brush parts, pail of water | Multiple | Low | Parts |
| Schügerl et al. [15] | 46 % | 75 % | 0.57 | cars | Single | High | Whole |
| Ours (video 2) | 42 % | 72 % | 0.53 | body parts, brush, golf club, towel | Multiple | Low | Parts |
| Schügerl et al. [15] | 32 % | 70 % | 0.44 | persons | Single | High | Whole |
| van de Weijer and Schmid [19] | 55 % | n/a | n/a | birds | Single | Low | Whole |
| van de Weijer and Schmid [19] | 43 % | n/a | n/a | soccer teams | Single | Low | Whole |

Table 6. Comparison of results with similar applications.

## 5 Other Approaches

Besides trying spatial transforms to calculate the centroid instead of the mean (as discussed earlier), we experimented with a few other approaches that produced inferior results. These approaches are listed in this section, as they can help with better understanding of the challenges.

### 5.1 Deblurring

Camera movement causes blurring. We deblurred the test images in an unsuccessful attempt to improve the detection rate. A blurred image can be approximated by [10]:

$$g = Hf + n$$

where,

$$g = \text{blurred image} \tag{8}$$

$$H = \text{distortion operator} \tag{9}$$

$$f = \text{original image} \tag{10}$$

$$n = \text{noise introduced by camera} \tag{11}$$

We experimented with deblurring the test image using the MATLAB deblurring functions [10]: deconvwnr (Wiener filter), deconvreg (regularized filter), deconvlucy (Lucy–Richardson algorithm [12]), and imfilter (coorelation filter). Deblurring is an iterative process requiring knowledge of the distortion operator (H), which is unknown. However, there are a few standard MATLAB distortion operators that are used for deblurring, which we also used: motion (approximates the linear motion of a camera), Gaussian (Gaussian low pass filter), average (averaging filter), and disk (circular averaging, or pillbox, filter).

## 5.2 Number of Views

We looked at the number of views included in the Parts Database. As shown in Table 7 and Figure 7, the results improved as we added views until there were 39 views in the database and then the results began to deteriorate.

| Views | 2 | 5 | 11 | 23 | 39 | 78 |
|---|---|---|---|---|---|---|
| Recall | 9 % | 25 % | 36 % | 46 % | 49 % | 45 % |
| Precision | 18 % | 39 % | 53 % | 72 % | 71 % | 67 % |
| F1 | 0.12 | 0.3 | 0.43 | 0.56 | 0.58 | 0.54 |

Table 7. Experimental results for various numbers of views in the Parts Database (MatchThresh = 90 %).

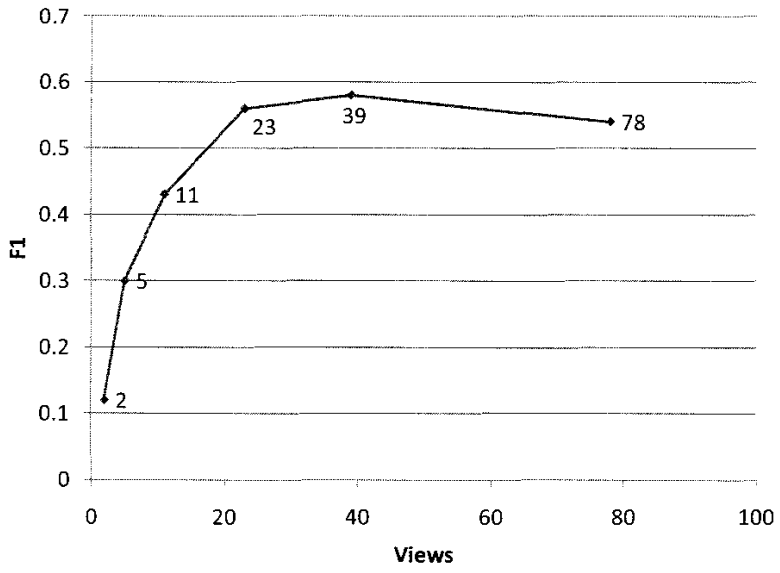


Figure 7. Experimental results for various numbers of views in the Parts Database (MatchThresh = 90 %).

### 5.3 Centroid Calculation

Instead of using the mean of the detected keypoints of a part as the centroid, we also investigated using the single detected keypoint with the shortest Euclidean distance between the SIFT keypoint in the test image and the SIFT keypoint in the Parts Database. However, the results were worse. We think this may be due to the fact that using a mean eliminates the false-positives that occur when a keypoint detected near the edge of a part falls outside the bounding polygon.

### 5.4 Edge Threshold

We studied using different values of the vl_sift edge threshold parameter. The edge threshold eliminates peaks of the DoG scale space whose curvature is too small, since such peaks yield badly localized frames. Increasing edge threshold increases the number keypoints detected. As Table 8 and Figure 8 show, the optimum results, as measured by F1, are achieved when the edge threshold is 20.

| Edge Threshold | 10 | 15 | 20 | 25 | 30 | 40 | 50 | 75 | 100 | 200 |
|---|---|---|---|---|---|---|---|---|---|---|
| R | 44 % | 48 % | 49 % | 48 % | 48 % | 48 % | 48 % | 48 % | 47 % | 47 % |
| P | 68 % | 70 % | 71 % | 71 % | 70 % | 71 % | 72 % | 73 % | 73 % | 72 % |
| F1 | 0.54 | 0.57 | 0.58 | 0.57 | 0.57 | 0.57 | 0.57 | 0.57 | 0.57 | 0.57 |
| Keypoints | 1202 | 1298 | 1340 | 1361 | 1380 | 1396 | 1397 | 1397 | 1397 | 1399 |

Table 8. Experimental results for various values of vl_sift edge threshold.

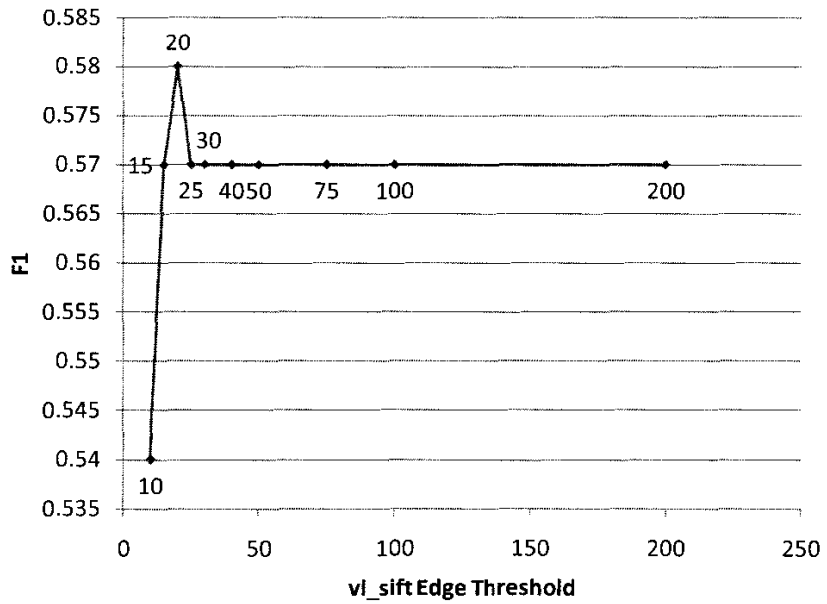


Figure 8. Experimental results for various values of vl_sift edge threshold.

Table 8 illustrates that the number of keypoints detected does increase as edge threshold increases, but levels off around an edge threshold value of 40.

### 5.5 Conclusion

In this paper, we have presented a novel efficient algorithm for locating and tracking object parts in low resolution videos using Lowe's SIFT keypoints [9] with a nearest neighbor object detection approach. We illustrated using experimental data that this algorithm can achieve a recall of 49 % with a precision of 71 % and an F1 of 0.58. Table 6 shows that these results are better than those achieved in less demanding applications.

The next step in our research is to use this algorithm to identify and track the trajectories of objects (e.g., scrub brush, and pail of water) and major body parts (e.g., hand, head, and body) in instructional videos. Then, we can use a Machine Learning technique, such as Hidden Markov Models, to recognize the trajectories as proto-symbols, where each proto-symbol represents one type of motion (e.g., picking up a scrub brush, dipping the scrub brush in a pail of water) [6]. And finally, we can use the proto-symbols to teach a humanoid robot, through RbD, to perform the task that is demonstrated in the instructional video.

Although the object detection performance of our algorithm compares favorably with current technology, it is still much lower than that of a human. We plan to overcome that limitation by taking advantage of the fact that we are using a video and not still frames. Thus, we can predict the $x$-$y$ coordinates of points in frame $n + 1$ from the trajectory of the points in frame n. Using this method, we can find the best-fit trajectory from the noisy trajectories that will be obtained from our algorithm.

## Bibliography

[1] Billard, A., Calinon, S., Dillmann, R. and Schaal, S. Robot programming by demonstration, in: *Handbook of robotics*, edited by Siciliano, B. and Khatib, O., Berlin, Springer, Berlin, 2008, chapter 59.

[2] eHow.com home page. `www.ehow.com`, accessed 27 January 2010.

[3] Ferrari, V., Tuytelaars, T. and Van Gool, L. Integrating multiple model views for object recognition, *Proceedings of IEEE conference on computer vision and pattern recognition*, Washington, DC, USA, June 27–July 2, 2004, 105–112.

[4] Hu, W., Tan, T., Wang, L. and Maybank, S. A survey on visual surveillance of object motion and behaviors, *IEEE Transactions on Systems Man and Cybernetics Part C Applications and Reviews,* **34**:**3**, 2004, 334–352.

[5] Inamura, T., Okadab, K., Tokutsub, S., Hataob, N., Inabab, M. and Inoue, H. 2006. HRP-2W: a humanoid platform for research on support behavior in daily life environments, *Proceedings of 9th international conference on intelligent autonomous systems*, Tokyo, Japan, March 7–9, 2006, **57:2**, 2004, 145–154.

[6] Inamura, T., Toshima, I. and Nakamura, Y. Acquiring motion elements for bidirectional computation of motion recognition and generation, in: *Experimental robotics*, edited by Siciliano, B. and Dario, P., Berlin, Springer-Verlag, **8:5**, 2003, 372–381.

[7] Jensfelt, P., Ekvall, S., Kragic, D. and Aarno, D. Integrating SLAM and object detection for service robot tasks, *Proceedings of IEEE/RSJ international conference on intelligent robots and systems, workshop on mobile manipulators: basic techniques, new trends and applications*, Edmonton, Canada, August 2–6, 2005.

[8] Leibe, B. and Schiele, B. Scale-invariant object categorization using a scale-adaptive mean-shift search, *Proceedings of German Association for Pattern Recognition (Deutsche Arbeitsgemeinschaft für Mustererkennung e.V., DAGM)*, Tübingen, Germany, August 30–September 1, 2004, 145–153.

[9] Lowe, D. Object recognition from local scale-invariant features. *Proceedings of the international conference on computer vision*, **2**, 1999, 1150–1157.

[10] The MathWorks home page. MATLAB version 7.4.0.287 (R2007a), `www.mathworks.com`, accessed 27 January 2010.

[11] Online Media Technologies Ltd. home page. `www.avsmedia.com`, accessed 27 January 2010.

[12] Richardson, W. Bayesian-based iterative method of image restoration, *J. Opt. Soc. Am.*, **62**, 1972, 55–59.

[13] Savarese, S. and Fei-Fei, L. 3D generic object categorization, localization and pose estimation, *Proceedings of IEEE eleventh international conference on computer vision*, Rio de Janeiro, Brazil, October 14–20, 2007, 1–8.

[14] Savevid.com home page. `www.savevid.com`, accessed 27 January 2010.

[15] Schügerl, P., Sorschag, R., Bailer, W. and Thallinger, G. Object re-detection using sift and mpeg-7 color descriptors, *Proceedings of international workshop on multimedia content analysis and mining*, Weihai, China, June 30–July 1, 2007, 305–314.

[16] Shakhnarovich, G., Darrell, T. and Indyk, P. *Nearest-neighbor methods in learning and vision: theory and practice*, Cambridge, MA, USA, MIT Press, 2005.

[17] Smeaton, A. F., Over, P. and Kraaij, W. Evaluation campaigns and TRECVid, *Proceedings of the 8th ACM international workshop on multimedia information retrieval*, Santa Barbara, CA, USA, October 26–27, 2006, 321–330.

[18] Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B. and Van Gool, L. Towards multi-view object class detection, *Proceedings IEEE conference on computer vision and pattern recognition*, New York, NY, USA, June 17–22, 2006, **2**, 1589–1596.

[19] van de Weijer, J. and Schmid, C. Coloring local feature extraction, *Proceedings of the european conference on computer vision*, Graz, Austria, May 7–13, 2006, 334–348.

[20] Vedaldi, A. and Fulkerson, B. *VLFeat: An open and portable library of computer vision algorithms*, `www.vlfeat.org`, accessed 27 January 2010.

[21] Zickler, S. and Efros, A. Detection of multiple deformable objects using PCA-SIFT, *Proceedings of AAAI conference on artificial intelligence*, Vancouver, British-Columbia, Canada July 22–26, 2007, **22**, 1127–1132.

Received January 13, 2011.

**Author information**

David O. Johnson, Computer Science Electrical Engineering, University of Missouri – Kansas City, 5110 Rockhill Road, Kansas City, MO, 64110, USA.
E-mail: `davidojohnson@aol.com`

Arvin Agah, Electrical Engineering & Computer Science,
University of Kansas, 1520 West 15th Street, Lawrence, KS, 66045, USA.
E-mail: `agah@ku.edu`