Tracked-base semi-supervised learning on camera-based system

Phumchanit Watanaprakornkul
Stanford Univerisity
Stanford University. Stanford, California 94305
yiam@stanford.edu

Abstract

One problem in supervised learning is that we need a lot of data in order to get a reliable classifier. There are many researches into semi-supervised learning to solve this problem. Many of those can be applied in object recognition and most of them take advantage of cues that come from specific type of input images. This paper will deal with input images from a video and use the fact that an object cannot just disappear and pop out at other place in the frame as our cue. We can track object in a video and label them accordingly to generate labeled data for traditional supervised learning algorithm. This semi-supervised learning method using track information has already been done on data from laser range finder and yield satisfying performance comparable to that of the traditional supervised learning with many times amount of labeled training data. The goal of this project is to extend such result to camera-based system. Our approach is doing background subtraction to filter out stationary objects that we do not care about, doing segmentation on frames to extract meaningful objects and then applying tracking algorithm to labeled more data accordingly. And finally, we could use the extra labeled data for traditional object recognition classifier.

1. Introduction

Supervised learning object recognition algorithm has one main weakness: the need for large amount of labeled training data. So, we there are a lot of research going into semi-supervised learning, which require only a few labeled data and a large amount of unlabeled data for the training process.

Semi-supervised learning has been done before and there are many ways to do it such as using generative models to generate more training data, self-training, co-training, etc. [8]. In our case, we are doing semi-supervised learning on object recognition. There are many ways to do so. Most of them rely on using some clues in special type of data to generate more labeled data for training process. (such as [9]) In our case, our data is a video. We are going to do semi-supervised learning by generating more labeled data using tracking information in a video and a starting point from a few manually labeled ones.

We can label an object of one type in one frame then use a model-free segmentation and tracking to find the position for the same object in the next frame. We then label the object in the next frame of the same type and so on, resulting in more labeled data. In this way, we get enough training data for traditional supervised object recognition.

The idea of using tracking information to generate more labeled data for supervised learning is already done in [1]. However, the data from LIDAR sensor used in [1] is very different from frames that we get from normal camera. We cannot apply exact same process because the segmentation and track classification will be very different. In addition, camera is cheaper than LIDAR sensor.

2. Background / Related works

Due to the difficulty to acquire sizeable labeled data, there is a lot of research on semi-supervised learning [8]. We are exploring a few general approaches for semi-supervised learning here. Firstly, the oldest one is the generative models ($P(x, y) = P(y) P(x \mid y)$). With it, we only need one labeled example per component. There are many variance of it and many of them use EM to learn the necessary parameters. Secondly, we can do clustering on every data point and label each unlabeled data point in the cluster by the vote of labeled data inside that cluster. Thirdly, self-training suggests that we start training with the very small set of manually data point that we have. Then use the trained classifier to classify the unlabeled data, pick a few classifications with very high confidence and label them accordingly. We then retrain the classifier with the labeled data (including the newly labeled ones) and so on.

Semi-supervised learning on object recognition has been done before. Object detection on aerial imagery done in [9] is a variation of self-training. It has special labeling strategy based on context of an object. This context cue, specific to aerial imagery, contains surrounding regions and enclosed regions. Semi-supervised learning has been applied on normal image as well. In [10], it is basically a variation of

generative model and is done using conditional random field spatial integration of local features and segmentation cues. These semi-supervised learning approaches have comparable performance to supervised learning in their own specific type of data.

This paper is exploring semi-supervised learning on video stream. The idea of using tracking information to generate labeled data for supervised learning is proven to be useful in [1], where the tracking is done on 3D range data collected by LIDAR sensor. The project collect range data from streets and compare track-based semi-supervised learning with the traditional supervised learning algorithm. Given a few manually labeled training data, the project apply EM based algorithm on tracking information of 3D range data, match them with the snapshot of other time frame and generate more labeled data. The result of the project is that while the supervised learning score around 99% accuracy, the tracked-base semi supervised learning score 95% with the tracking part label 81.8% of the extra training data correctly. From this good result, we are applying the same idea on our project.

3. Approach

The main part of this project is to generate labeled training data for the object classifier from small about of manually labeled data. This data are a part of a frame from a video.

There are two ways to implement this. One approach could be that we can specify positions of our objects of interest in the video and track them. However, our model free segmentation and tracking have limitations which can cause us to lose the object sometimes. In addition, some we cannot add more training data in some cases such as the labeled pedestrian just walk under a tree within a few frames. So, we might pick a bad labeled data to track.

The better approach is from the training video, we extract a list of objects and their positions in each frames. Track objects over frames and label them whether they are the same object or not. Then, we output all objects we found. The output could be large. We can manually pick a few objects from the output by choosing ones that appear in many frames. This way, we get more data for the classifier. Moreover, it is easier to correct any tracking or segmentation error if it arises.

There are mainly four steps for doing semi-supervised learning method using track information. We first subtract the background from our video. Then, we do segmentation to extract objects of interest from each frame. We then do tracking to map the same objects of interest across frames. These three steps are model free and they will give a list of tracked objects for us to label. The last step is classification.

3.1. Background subtraction

We first have to filter out background which we are not interested in. This make segmentation step easier. In general, objects that we track tend not to clutter together and background separated each object from each other. If the background does not, it is still easier to filter the background out to lower complexity of our problem. This can be done by standard background subtraction algorithm.

We use OpenCV implementation of Gaussian mixture based background subtraction algorithm described in [3]. This algorithm aims to filter out non-moving part of a frame as background. It works in most cases. For a frame, all frames before it are included in our Gaussians and the later frames are more important than the former. This handles brightness change, and small movement of the camera.

3.2. Image Segmentation

After removing the background, we then have to divide the frame into segments/cluster of possible objects. We do this on the foreground that we get from background subtraction.

The first approach for this model-free segmentation is to use mean-shift. However, it is difficult to assign a correct window size for mean shift and the algorithm has sizable running time. Given that we have many frames to process and each frames has millions pixels, mean shift is too slow.

We have a simpler and faster method that works well for our problem. We can operate on the foreground mask, which is a matrix of bits, instead of the foreground image. And given that in general, our object of interest will not overlap each other. We can just group the connected pixels in the mask together as a cluster. We do this by using flood fill. This is a lot faster than mean shift. Sometimes, pixels of an object are not connected because our background subtraction is imperfect. We can fix this by applying Gaussian blur filter to fill the missing pixels in the foreground mask. Alternately, we can get the similar result by modifying flood fill to go to near pixels in some radius instead of only adjacent pixels. From our tests, radius $r = 3$ works well with OpenCV Gaussian mixture based background subtraction.

3.3. Tracking

In this step, we add the tracking information into each segment in the frame, determining that this segment come from which segment in the previous frame.

We start with a simple and fast approach similar to segmentation step. For each cluster of pixels representing an object of interest, we use the center of the cluster (the arithmetic mean) to represent its position. We then match the cluster with the nearest cluster in the previous frame

(using distance between centers as a criteria), labeling them as the same object. There are optional constraints to enforce here such as the mapping of a position of an object from one frame to another frame has to be one to one (if we ignore the case that segmentation sometimes fail when more than one objects are colliding or moving to the same location). Also, there are cases where this would mistakenly match two different objects together. For example, when one object moves out of the scene at the same time the other one moves in. We fix this by setting a threshold for the distance we consider.

Another complication is that the background subtraction and segmentation is not perfect, so sometimes we lost the object in a few frames. For example, two pedestrians walk in opposite direction overlap each other in a few frames before walking away. Our segmentation cannot handle the overlap, so we lost the objects in that frame. To fix this, we look back into more frames in the past and match objects based on distance of the clusters and the distances in time frame.

Another viable approach is that we use Kalman filter to predict the most possible position of an object in one frame in the next frame. Then search for an object in the next frame around the predicted position.

3.4. Object Classification

We then use the generated training data labeled by tracking information to train our usual object classifier. This part is not the main point of the project. However, we are trying two object recognition algorithms.

The first one is Scale-Invariant Feature Transform (SIFT) [11]. SIFT is one of the most popular algorithm for object recognition. However, SIFT is generally used to match an exact object from image to another image which that object could change scale, orientation, and affine shape adaptation. Our object classifier, on the other hand, wants to classify objects into more general categories such as classifying pedestrians and bikers on the street. Therefore, we have to modify SIFT in some degree.

From each training image in a class, we extract SIFT keypoints associate with it and keep it per training image. In classification process, we try matching each of these training images to the keypoints in the test image (frame from video). We score each match by the number of keypoints used, orientation, and how far keypoints spread. The we classify the match with score higher than some threshold as belonging to a specific class.

The second algorithm we use for classification is the one more generally used to classify part of images in more general categories more suitable to our job than SIFT. We are using Discriminatively Trained Part-based Model as described in [4].

4. Experiment

4.1. Dataset

There are several limitations of our approach. Since the background subtraction, segmentation, and tracking steps have to be model-free, our approach could only work on a set of object. Firstly, by using background subtraction, we will not be able to use any object in the background for training the classifier. We generally classify non-moving objects as background, so we can only consider moving objects that come in and out of our camera view. Note that even if we consider non-moving objects, they would stay the same for every frame in any case. So, the tracking part will not be able to generate more labeled data for supervised learning.

Secondly, our model-free segmentation, especially the one that does not need any parameter, will extract objects from an image in some specific way. The simple flood fill method described above will extract all connected foreground as one object. If there is a pedestrian, it will extract the whole pedestrian. It cannot differentiate between head, leg, and any other part of an object.

Figure 1: image, foreground mask, and foreground

From figure 1, given that parts of the biker after background subtraction are all connected, simple flood fill method will always extract the whole part of the image as a biker and cannot extract any specific as head, wheel, leg, etc. Therefore, objects that we could apply our approach for semi-supervised learning has to be the whole object that every connecting part of it is continuously moving.

From these limitations, cars, bikers, and pedestrians on the street seem to be a few of the best options. For simplicity, we prefer that the camera is not moving. Our dataset is a video of overhead view from Hoover tower to the fountain in front of it. We are interested in recognizing moving objects, in this case, pedestrians and bikers.

4.2. Background subtraction stage

The OpenCV mixture of Gaussian background subtraction works well in most cases. However, it creates some problem as well. When our object of interest stops moving for a few frames in the video, it disappears into the background. We will detect it when it starts moving again and we will not be able to associate the objects before and after the stop to be the same object. See figure 2 as an example.

Figure 2: three sample frames of pedestrians with their corresponding foreground mask. Each sample frame is 100 frames further form the next one.

Although we could be able to fix this issue by using tracking information to keep stopping object in the foreground, we can just ignore it because we already generate enough labeled data and we can pick other tracked object anyway. Moreover, the unmoving object that fades into background will stay in the same orientation and will not help the classifier even if we generate more labeled data out of it anyway. Therefore, there is no need to improve on this.

4.3. Image segmentation stage

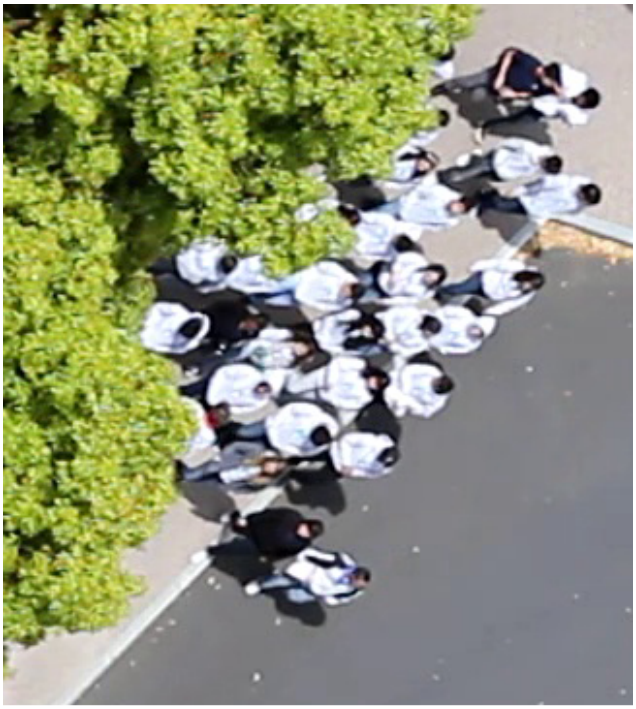
As we stated in the dataset section, our model-free segmentation will extract objects from an image in a specific way. In case of simple floodfill approach, the whole connected chunk of moving objects will be extracted as one object and we could only categorize pedestrians, bikers, and similar general objects. However, if we use mean-shift, it is possible to configure window size to extract parts of the general objects such as heads and legs of pedestrians, wheels from bikes, etc. (However, this won't be model free anymore.) The other difference between simple floodfill and meanshift is that simple floodfill is a lot faster and way easier to implement. Given that we have already done background subtraction which enable floodfill to work and we are interested in pedestrians and bikers, it is suitable to go with floodfill approach.

The limitation of our model-free segmentation approach is that it cannot deal with overlapped objects such as a group of pedestrians walking together. It will cluster the whole group as one object. Therefore, our method is ineffective when there are a lot of object of interest in a frame at once and they are potentially overlap. In our test video, there is only one group of pedestrians walking across the scene, so we can just pick other pedestrians as training data. See figure 3 on the next page for this case.

Note mean shift might be able to handle a group of pedestrians in the foreground; however, it require a lot of manual configuration to avoid making the resulting clusters being several heads and several shirts instead of separated pedestrians.

4.4. Tracking stage

The simple method of using distance between centers of object works better than we expected. It is good enough that there is no noticeable benefit to switch to use Kalman filter to predict the position of the object in other frames. In total, the background subtraction, segmentation, and tracking on 20,000 frames of the input video take around five hours on common machine.



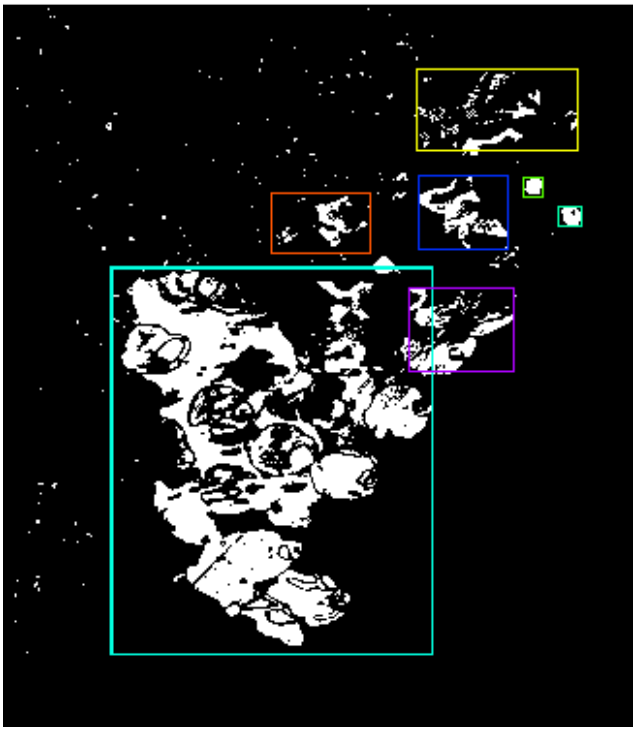


Figure 3: a group of pedestrians, foreground mask, and result from simple flood fill segmentation. We can see that the result is not suitable for out object recognition.

However, there are some problems. In the test video, we have some noises in a cluster of swirling tree leafs, but those are not in the region of our objects of interest. In addition, we cannot handle the case that two objects move along opposite directions, then occupy the same position, and move past each other. In rare cases, we mislabel the object after they disengage.

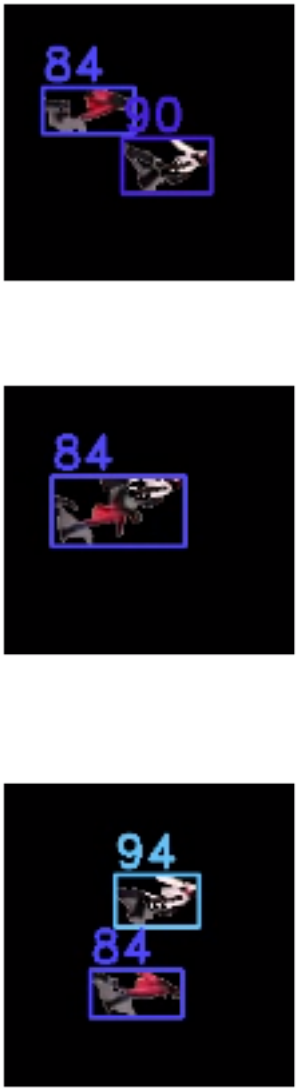


Figure 4: two pedestrians walk in opposite direction then disengage. Our tracking algorithm outputs a wrong label for an object after disengagement. This rarely happens.

4.5. Final classification

We process the first 20,000 frames from the video. Our tracker output 1839 possible objects of interest. They mostly consist of pedestrians, bikers, cars, and moving

leaves. The number of labeled data of each object varies from 2 to around 700. Pedestrian tend to generate more data because they move slowly, so one pedestrian appear in many frames before the person walks out of the camera view. On the contrary, moving leaves has few frames. (We ignore them anyway since we only consider pedestrians and bikers.)

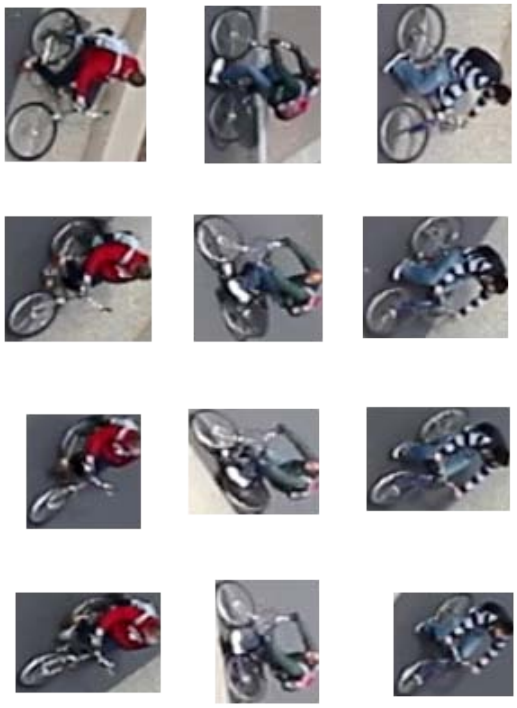


Figure 5: samples of labeled training data generated from our tracked-base approach. We use these to classify bikers. Since we manually label 7 bikers, the whole data we use to train have 7 columns and hundreds rows.

The point of semi-supervised learning is to pick a few labeled training data manually. So, we pick 10 objects of pedestrians and 7 objects of bikers, which give us 3488 images of pedestrians and 1394 images of bikers. These are enough training data. So, we use them on our classifier.

The first object recognition algorithm we consider is our modified SIFT. We modify the code from SIFT library by Rob Hess. It is very slow and does not work very well. It take about a minute to run test on a frame of size 1900 x 1088 pixels trained by the generated labeled data above. We cannot find a good score threshold to return a meaningful match. We can return the top score match; however, they tend to belong to the same object that we train with and it only happens when the same object appear on the test frame. (We can only detect a pedestrian in a test frame if the same pedestrian wearing the same outfit exist in our training set. See figure 7 on the next page.) This is due to the fact that different training images generated has a different number of keypoints. We also found that normalize the number of keypoints along training images or trying to combine keypoints from different training images does not increase any performance of the classifier. We believe that the problem about only able to recognize the exact same object in the training set is due to how SIFT work. It is too specific for our job and categories of objects do not belong to SIFT invariance. In addition, supplying the same object in several frames whose difference are only scale and orientation cannot give any more information to SIFT features because SIFT is invariant to scale and orientation itself. Therefore, we should not use SIFT with our semi-supervised track-based approach.

SIFT is not quite compatible with the way we label training data; however, part based classifier as described in [4] works fine with our approach. It takes a couple days to train part-based models; however, the classification process of test frame is fast and yield good performance. We can detect most objects in the test video.

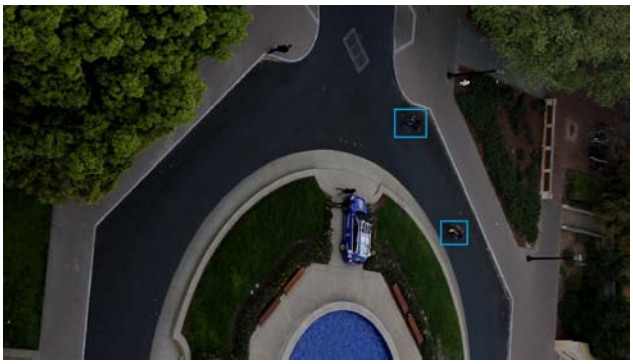


Figure 6: a sample output of object recognition for bikers using part based model. The boxes indicate bikers location.

Note that sometimes, with a specific set of generated labeled data that we select, the part-based model learning algorithm fails overlapping condition (see MATLAB code for more information).

5. Conclusion

In this paper, we see that we can apply tracked-base semi-supervised learning for object recognition on a video stream. We achieve this by filter out background from every frame in the video, apply segmentation method that is appropriate to the video we have and the objects we want to recognize, then track and label each object in each frame accordingly. From here, by manually picking only a few objects, we end up with a thousand labeled image for training.

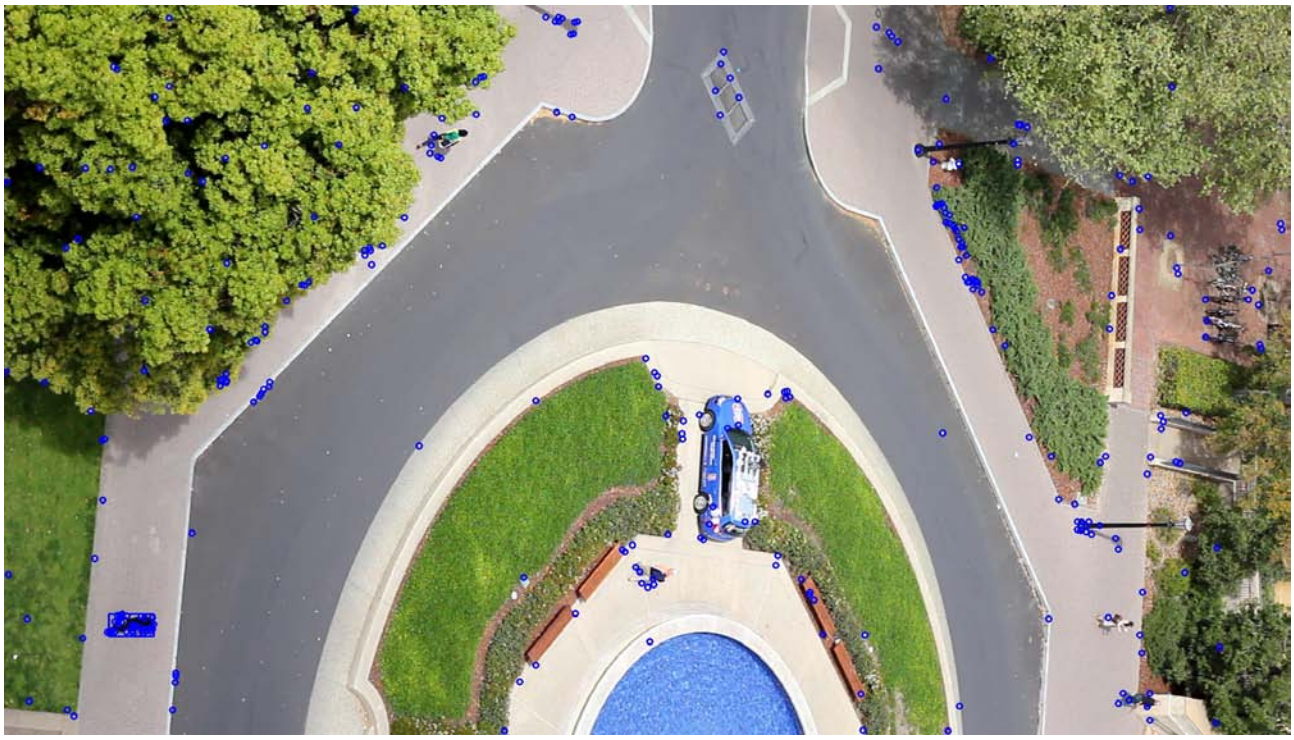


Figure 7: Object recognition for pedestrian on one frame using SIFT descriptor and classifier. The blue circles are key points. We can see that it recognize only one pedestrian on the bottom left, which that specific pedestrian belongs to labeled training data. It does not recognize the other pedestrians.

There are limitations for our approach of the tracked-base semi-supervised learning. One is that the types of objects we can classify are limited to moving object (because of background subtraction and the fact that non-moving object will not give us any different training image generated by segmentation and tracking). And given that the phases use to generate more labeled data have to be model-free, we are limited to whole objects that consist of the every moving part. We cannot recognize only a part of moving object. (We can recognize pedestrians, but not heads of those pedestrians.) We can configure the segmentation stage (see section 3.2) to make our approach be able to recognize different types of object; however, it will no longer be model free and the label data generation will be more complicated.

Other than the limitation on objects that we are trying to do object recognition, we are also limited to a specific type of descriptor and classifier. As we can see in section 4.5, some descriptor or classifier such as SIFT is just not suitable to use with tracked-base semi-supervised learning approach. From the way we generate more labeled data, we will have only a few objects, which we select manually, but we have many frames of those objects in many different orientation (and sometimes different scale). Therefore, the descriptor and classifier that will benefit from tracked-base semi-supervised learning approach must not have orientation and scale invariant (otherwise the extra labeled data that we generate will mean nothing). The deformable part model we tried is one of the models that work well.

The next step for this project is making tracked-base semi-supervised learning work on moving camera. Moving camera will enable us to apply track-based method on non-moving object because the camera can capture the stationary object at different perspective and label them the same object. However, the process will be more complicated than the steps described in this paper. Background subtraction might not be a good idea anymore if we want to extract non-moving objects in the background. Even if we want to subtract background, we need to use a different and more complicated model of background subtraction to handle the movement of the camera. The simple approaches in segmentation and tracking will no longer work, so we will have to use mean-shift and Kalman filter. There are possible future works.

References

[1] Alex Teichman, and Sebastian Thrun. “Tracking-based semi-supervised learning”.

[2] Thanarat Horprasert, David Harwood, Larry S. Davis. “A Robust Background Subtraction and Shadow Detection”
[3] P. KaewTraKulPong and R. Bowden, “An improved adaptive background mixture model for real-time tracking with shadow detection", Proc. 2nd European Workshop on Advanced Video-Based Surveillance Systems, 2001. < http://personal.ee.surrey.ac.uk/Personal/R.Bowden/publications/avbs01/avbs01.pdf >
[4] P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, No. 9, September 2010
[5] N. Martorana, L. Masi, M. Meoni, “Kalman and Condensation in video-tracking”. Universita Degli Studidi Firenze
[6] Gary Bradski and Adrian Kaehler. “Learning OpenCV” O'Reilly Media, 2008.
[7] R. Fergus, P. Perona, A. Zisserman, “Semi-supervised learning and recognition of object classes”.
[8] Xiaojin Zhu, “Semi-supervised learning literature survey”. University of Wisconsin-Madison, 2008
[9] Jian Yao, Zhongfei Zhang. “Semi-supervised learning based on object detection in aerial imageery”
[10] Peter Carbonetto, Gyuri Dork´o, Cordelia Schmid, Hendrik K¨uck, and Nando de Freitas. "A Semi-Supervised Learning Approach to Object Recognition with Spatial Integration of Local Features and Segmentation Cues"
[11] Lowe, D. G., “Distinctive Image Features from Scale-Invariant Keypoints”, International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.
[12] Peter S. Maybeck, “The Kalman filter: an introduction to concepts”. Stochastic models estimation and control. Vol 01. Ch 1.3. 1979

Appendix

This project extends the result of tracking-based supervised learning done on LIDAR data[1] to data from normal camera. This project is supervised by Alex Teichman. The video used in the experiment is provided by Alex as well.

We use a few open source packages in this project: OpenCV. SIFT library by Rob Hess from Oregon State University. And Discriminatively Trained Deformable Part Models by Felzenszwalb from Brown university.

Future Distribution Permission

The author of this report gives permission for this document to be distributed to Stanford-affiliated students taking future courses.