**Introduction**

You walk on the street in Manhattan where the most crowded place in the world. Your eyes suddenly land on a person who you think attractive, and you can’t stop keeping your eyes on him/her as this person is moving in the crowds. This is a simple explanation of object detection and tracking performed by our brains. These two are totally separate in respect of the way our brains process, but both serve to achieve relatively the same interest. In computer vision, object detection relies on recognizing the object of interest by searching for distinct features, and as for object tracking, those features assist us to identify how the object moves. Two examples we have covered in class are local feature detection by SIFT and optical flow. Local feature detection by SIFT informs us where the object is, because we expect the object should contain those local features. Though optical flow is commonly used for estimation of movement rather than object tracking, both movement estimating and object tracking are revealed by examining how local features behave. That’s say, to solve optical flow problems, we need to how each pixel behaves or a group of pixels as “corner” behave. One aspect of advance our brains have in tracking is that we instantly fit a contour for the object, but not perform a lazy job that everything in our eyes is within a bounding box. Philosophically or theoretically, it is hard to debate that our brains track objects without first finding a bounding box with an ideal size. But we do further draw the precise contour that fits the object. The benefit of contour fitting can be understood in two respects.

Previous works have demonstrated the state of art technique of object detection and tracking. YOLO XXX. On the

**Method**

YOLO

**Active Contour**

**External Energy |** We consider three sources from the image to define external energy. First, pixel value. The object of interest is assumed to have distinct pixel values from the background. Second, edge. The edge should be conserved between the objective and the background. Third, corners. The pedestrian is assumed to have different texture that can be defined by corners from the background. We give different weights to these three terms to highlight the source of external exergy. For example, higher weight of edge indicates larger external energy source from edge.

**Energy Points and Clustering |** Energy points are identified by calling local minima from external energy map. The local minima should satisfy minimal values in all orientation. Once we identified key energy points, we use position of point (x and y) as 2D-feature vector and cluster them into groups by k-means. We the set the centroids of cluster and between clusters as the initialization points and draw circles based how far the centroid to the furthest point. Those circles are our initial contours.

**Contour Fitting |** the active model is adopted from the algorithm proposed by Kass et.al. [1]. In order to get higher performance, we set up the fitting in 4 iterations. In the first iteration, the model starts from 3 arbitrary values respect to in the Eq.2. In the second iteration, we fix and decrease while increasing . In the third iteration, we further increase but fix . In the final iteration, we decrease . We reason allowing the model fluctuation between elasticity and stiffness may return a better contour. In fact, choosing 3 arbitrary values for the three terms didn’t result a good contour fitting. However, fluctuation of parameters allows the model to find optimal solution. Though we don’t have proof that these 4 iterations work better than one single iteration, we found these 4 iterations return better performance by experimenting. We also experimented reiterate iteration 3 and 4 until the total external energies of contour converges, but it didn’t improve a significant outcome.

Eq.2.

**Mask for Evaluation |** To evaluate how well contour fits the object, we convert contour into the mask. Mask pixel will be 1 and background pixel turns to be 0. We compare the true value and the predicted value for each pixel in the image and evaluate the performance by precision and recall of 1s, because we expect the model has higher precision on 1s (object) while retrieving as many as 1s.

**Experimental Results**

We first tested if one active contour model can fit the whole object (the pedestrian). We initialized the active model at the central pixel of the local and drew circles with different radiuses (100, 150 and 200 pixels respectively). The active model with a small radius shrunk to a curve that fits the object. Active models with larger radius start to form a shape that can partially mask the object (Fig. 1). As radius increases, more object pixels are retrieved by the model (recall), and meanwhile, precision of the model is also increased. However, one active contour centered at the local image is unable to cover the majority of the object (recall under 0.5 in all three tests. See Table 1). However, increasing radius also causes the model to retrieve irrelevant background pixels. This led us to employ multiple active contours to fit one object, and each contour is aimed to fit a part of the object.

|  |  |  |
| --- | --- | --- |
|  |  |  |

One previous study using deformable patches with level set method suggests outperformance by segmentation into patches [2]. Therefore, we consider fitting contour for the object by fitting multiple contours for sub objects. In order to find good initializing spots in the image, we find key energy points in the image. The reason behind is that those key energy points should be resulted from the distinctness of the object from the background. Therefore, regions containing key points can be good places for initialization. To retrieve back the object as whole, we also need to connect these local regions. Thus, we finished the pipeline in 5 steps: 1) calculating external energy of the image; 2) finding local maxima and thresholding to identify key energy points; 3) clustering energy points into groups; 4) fitting active contours for each cluster and connected regions between any two clusters (Fig. 2).

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

We next tested the influence of the number of energy points and the number of clusters separately. Table 2 shows calling out more energy points doesn’t improve both precision and recall (See Table 1). Noticeably, precision is affected more than recall. This may be the reason that lower threshold returns more energy points contributed by background and the contour model starts to fit irrelevant regions. On the other hand, the number of clusters is critical for the performance. Increasing the number of clusters has bare impact on recall, suggesting how much relevant object regions are determined by how well energy points represent the object. The same as effect of energy points, increase of clusters reduces the precision. There may be two reasons: 1) clusters are dominated by energy points contributed by background; 2) over-clustering causes outlier energy points.

**Conclusion**

In this project, we first trained convolutional neural network (YOLO) that can accurately detect pedestrians. Data augmentation by XXX overcame the limitation of size of dataset. We successfully implemented YOLO to narrow down a local object region for active contour fitting. Our experiments showed the incapacity and poor precision of one active model for the object. Our key energy point clustering approach greatly improved the contour performance. We further experimented how the number of energy points and clusters affect the final contour, and we suggest choice of threshold for energy points and cluster number should be considered case by case. However, fewer energy points but mainly contributed by object and fewer clusters should return a relatively high performance, generally. This is because fewer energy points reduce the risk of including more irrelevant pixels and fewer clusters weakens contribution of energy points from non-objects. We also suggested allowing fluctuation of active model by multiple iterations can greatly improve overall performance.

However, we didn’t achieve more than 0.9 precision and 0.9 recall in this project. We consider future work can be done to improve our algorithm. First, YOLO XXX. Second, good energy points are determined how well the external energy map distinguishes the object from the background. Considering there are three weights in the external energy equation, we argue searching a good combination of energy source weight systematically can help us get as many energy points contributed by objects as possible. However, this should vary case by case. Third, removing outliers before clustering energy points can also reduce risk of initialization in a bad local region. Therefore, detecting outliers or clustering algorithms that are robust to outliers would be used for testing. Fourth, we also consider the initial shape of contour will influence the outcome. In this project, we initialize every contour by assuming it is a circle. However, parts of the object are different, and assuming parts of the object with certain shapes may also improve the result. For example, the walking legs can be thought as ellipse with a certain rotation. Finally, designing a systemic approach that active contour can iterate to the optimum not only produces stable outcome but also give us cues how good model is rather than evaluating by mask.

**Reference**

1. Kass, Michael, Andrew Witkin, and Demetri Terzopoulos. "Snakes: Active contour models." *International journal of computer vision* 1.4 (1988): 321-331.
2. Sun, Xin, et al. "Non-rigid object tracking via deformable patches using shape-preserved KCF and level sets." *Proceedings of the IEEE International Conference on Computer Vision*. 2017.