CS223B: Find Mii Challenge

Team: The Y-Nots

Sandeep Sripada
ssandeep

Venu Gopal Kasturi
venuk

Abstract

We describe the approaches taken to solve the 4 challenge problems: (a) Find this Mii, (b) Find 2 look-alikes, (c) Find n odd Miis out, and (d) Find the fastest Mii in the "FindMii" game. The approaches varied depending on the problem at hand and in this report, we would justify the reasons for the selection of a family of features and also explain the improvements implemented. We used HOG features and Boosted decision trees to train a new classifier, SIFT to match faces, optical flow to detect the direction and amount of motion, Mean shift clustering to group similar points together etc to successfully navigate the tasks.

1. Introduction

"Find Mii" is a game on Nintendo Wii Play and it basically involves identifying certain avatars (Miis) from a bunch of them, standing still or moving around in various styles. In the first problem, we would be given a reference Mii and the task is to identify the given Mii in the video. In the second problem, the task is to identify the most similar Miis. The Miis could be wearing different styles of clothing and could all be doing some activity rather than standing upright with good frontal face views. The third task is to identify odd Miis i.e. odd in their styles of shaking heads or footsteps. The final task is to find the fastest Mii i.e. from among the bunch find the Mii that does the common activity of either swimming or running the fastest.

The following sections would talk about each individual task in detail giving reasons on the choice of features and the additional improvements included.

2. Find this Mii

As mentioned above, this task requires the system to identify a given reference Mii from among a set of Miis under various environment changes. This task is further subdivided into 3 levels each with a different background.

- Level 1: Upright Miis arranged in rows.
- Level 2: Upright Miis moving on a vertical escalator.
- Level 3: Rotating Miis.

2.1. Features

As the Miis are rotating in one of the levels, it was required of the technique to be rotation and scale invariant. This is most important as matching with features that are rotation invariant is far more effective and quicker than looking at all frames and extracting windows and matching at various angles to the given reference image.

So, we used DoG feature detectors and SIFT descriptors [4] to match the given image and the frame in a video. This helped us in all the levels, as the scale and rotational invariance covered all scenarios presented in this task. The system computed the SIFT matches and returned a click with the frame and location information that had the maximum matches. As all the Miis are present in the video from the starting frame, we just needed to analyze the first frame. The resulting system was accurate and fast.

3. Find 2 look-alikes

This task requires the system to identify two look-alikes from among a set of Miis under various environment changes. This task is further sub-divided into 3 levels each with a different background.

- Level 1: Upright Miis.
- Level 2: Rotating Miis.
- Level 3: Swimming Miis.

3.1. Level 1

Features: To identify similar Miis, the first step was to identify faces. For this, we used a Haar-feature based cascade classifier [6] trained on Mii faces to recognize faces and then used the identified regions to calculate SIFT features. These features were then matched and the pair that had the largest match-ratio was selected as the final pair.

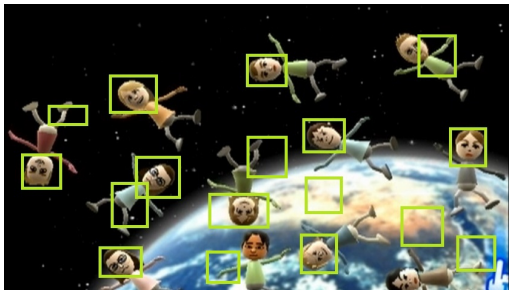

Figure 1. Classifier output after Non-maxima suppression.

We used a metric called 'match-ratio(mr)' to rank the pairs. The value is defined as follows: $mr(I_1, I_2) = matches(I_1, I_2)/min(\#descriptors(I_1), \#descriptors(I_2))$, where the numerator is the number of matches obtained after SIFT match and the denominator is the minimum of the number of descriptors obtained for each image.

3.2. Levels 2&3

Features: As the Miis in this level were not upright, a different kind of classifier was needed that would be able to recognize faces in a different angle/environment. To get this, we used a HOG [1] features based Boosted Decision Trees (BDT) classifier. The reason to choose HOG was to obtain scale and illumination invariance. Scale invariance is obtained by varying the sizes of the scanning windows and illumination invariance is obtained as HOG uses gradients. BDTs were used because of the low training time and effective parameter tuning to avoid overfitting to the training data.

The training data used for this classifier was generated from samples taken from the Mii videos. The positive samples obtained were rotated with various angles and placed as positive samples in the data set. The negative samples were also dealt with similarly. This helped in using the classifier in all cases i.e. side views, rotating etc. The output from the classifier is shown in Figure 1. The output of the classifier was also non-maxima suppressed to eliminate multiple outputs for the same regions.

From the output of the classifier, we use the Mean-Shift Algorithm [2] to obtain clustering. The clusters would group close by points to a single representative and this was used to look for potential matches for similarity. The clusters obtained using mean shift algorithm can be controlled by two parameters:

- Band width of the cluster: Increasing this would group far away points together.
- Number of neighbors within a cluster: Decreasing this would eliminate spurious sample points.

4. Find n-odd Miis

This task requires the system to identify n-odd Miis from among a set of Miis under various environment changes. This task is further sub-divided into 3 levels each with a different background.

- Level 1: Heads swaying from left to right.
- Level 2: Heads swaying from left to right.
- Level 3: Out of sync footsteps.

4.1. Levels

Features: This task was solved by applying optical flow on the frames to obtain the magnitude and direction of motion. The version of optical flow used is the Lucas Kanade Pyramiding [5] optical flow and the features to track were selected using Shi-tomasi [3] feature detection.

Once the direction and magnitude of the points was knows, some post processing was done to eliminate outliers and get an even output. The improvements include: trimming the magnitudes which are above a threshold from the mean, eliminating information from points that had an error above a certain threshold. After smoothing out the output, the points were binned into directional bins and the bin containing the majority of the points was considered the direction of movement. Using this, the odd Mii was identified and appropriate clicks were returned.

5. Find the fastest

This task requires the system to identify the fastest Mii from among a set of Miis under various environment changes. This task is further sub-divided into 3 levels each with a different background.

- Level 1: Running diagonally.
- Level 2: Running diagonally but with large number of Miis (smaller head size).
- Level 3: Swimming.

5.1. Levels

Features: This system started off by recognizing faces and calculating the Shi-Tomasi features in the given region of interests (ROIs) and applying optical flow as described above.

The output from the classifier was clustered using the Mean-Shift Algorithm and the obtained points were considered as centers of regions that were passed on to the optical flow module. The module then computes optical flow and fetches the point that has the maximum magnitude (specifying that the point is moving the fastest).

The improvements done here:

- As the points need to move in the same direction, a heuristic that only considers points within a delta angle difference i.e. almost paralel lines as contributors to the maximum. This eliminates outliers that occur with high magnitudes.
- Also regions that contribute small number of points are also eliminated.
- The error point were removed as before

The computations were carried out in sections of the video i.e. sets of frames were analyzed and the max from among them was taken. And the process is repeated over 4 such sets (under the assumption that within 2 seconds of the clip - the fastest would be in one of the frames - to reduce time complexity). The maximum of these maxima is then taken and output as the fastest point.

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[2] K. Fukunaga and L. D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

[3] C. T. J. Shi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[4] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[5] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *Imaging understanding workshop*.

[6] P. Viola and M. Jones. Rapid object detection using boosted cascade of simple features. *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

CS223B – Supplement

Sandeep Sripada (ssandeep), Venu Gopal Kasturi (venuk)

Task 1: Find this Mii

For the purpose of describing the features of both the reference and test image, we used **SIFT** (it uses **DOG** as the feature detector and describes the features). The features of the Mii image (reference) can match with many locations of the test image i.e. they would be clustered around a region if there is good match else the matches would not be clustered around a single region. So we need to identify if there is a dominant cluster of matches between the features of these two images. For this we have used **K-Means clustering algorithm** to cluster the matches on the test image. We have defined a dominant cluster as one which has more than 50% of the matches belonging to it and using that we return a click at the centroid of such a cluster.

Upon running the algorithm on various test videos we were able to fix k=3 for the k-means clustering. Too many clusters would result in very small clusters which might result in false positives as those clusters might contain very few matches. Too few clusters might result in the centroids being far off from the dense matches because of outliers and result in a click which is off by a large distance. Since SIFT is **scale and rotation invariant**, this worked for all levels.

Task 2: Find 2 look-alikes

The first step to recognize identical Miis is to effectively identify the faces. For this, we trained the classifier on a faces data set obtained by using the Miis at various angles (by programmatic rotation) and activities (swimming, floating etc.). The feature vectors used to represent the images were generated using: **Histogram of Oriented Gradients. Boosted Decision Trees** was used as the classifier for assigning probabilities to regions. The classifier used the scanning window mechanism to slide through the image to classify regions based on the training data. Since the size of Miis was around 40x40 in large cases and around 24x24 in case of smaller Miis, we used the appropriate sizes for training both scenarios and while classification we gradually shrunk the test image to obtain classification probabilities at various scale levels.

Level 1: Since the Miis were upright in this level, we used the Haar Classifier provided for the assignment and detected faces. Once face detection was done, we used SIFT to match the obtained results. Improvisation: Since matches may occur among non-similar faces as well, we came up with a metric to bypass this issue where higher match-ratio signifies a better match.

match-ratio(mr) = matches/min(ref image features, test image features)

Level 2&3: As the Miis in these levels are not upright, we used our custom classifier to recognize faces. After the face detection, we employed the **mean shift clustering algorithm** to obtain centroids of clusters which represented the center of the rectangles bounding the faces. Once the faces were obtained, **SIFT** was used to obtain similar Miis by **maximizing the sum of unique matches** between images and returning corresponding image pair. (**Improvisation:** unique matches is required as sometimes a single feature might match multiple features in another image which is not good).

Task 3: Find n odd Miis

Level 1: In this task, the Miis are rotating their heads side ways, left to right and vice versa. The odd-Miis are out of phase from the rest of the Miis. The procedure we have followed is to find out the good features to track using the **Shi-Tomasi algorithm** and track them using **optical flow**. But the problem in using this technique directly is that the features might contain points which are present as part of the

background too. Hence we eliminated such outlier points by using the fact that such points' optical flow values would be very less. By thresholding the magnitude of the flow vector of the features we were able to find out points which moved satisfactorily. The key is to compare two closely related frames of the video so that we can remove the background points, but at the same time these frames shouldn't be temporally far from each other as that would increase the error in obtaining the optical flow values. From statistical analysis on various test videos starting at different times, we found out that comparing two frames which are **8 frames apart yielded good results**. We then used **Mean Shift clustering algorithm** to cluster the features so that each cluster is formed by the features of the face of each Mii. The next step is to classify the Miis as clockwise/anti-clockwise rotation with respect to that image. This is done by looking at the dominant horizontal direction of each cluster of points and then returning clicks as the centroid location of the minority/odd Miis.

Level 2: This task is similar to the first one, but there are a lot many number of Miis and 3 odd Miis. The difficulty of this task lied in the fact that there were Miis on which there were no features to track. So sometimes (depending on the features selected), this task failed to output the right answer.

Level 3: In this task, we are required to identify the Mii which is walking in a different way. The advantage here is that all of the Miis are already present in the frame and their faces are in the upright position. This facilitates the use of a Haar classifier to identify all of the Miis in the first frame itself. After this stage, we used optical flow feature to analyze the motion of the different Miis. We obtained the **Region Of Interest** i.e. lower part of the Mii and obtained the optical flow vectors for this region and on analysis classified each Mii as left-footed or right-footed. After this we returned a click on the corresponding face of the Mii.

Steps of the task:
1. Identify the Miis by their faces
2. for each Mii recognized identify the lower body of the Mii
3. Select Shi-Tomasi features to track
4. Apply Pyramidal Lucas Kanade Algorithm to track these features
5. Analyze one half of the region of interest to classify as left-footed / right-footed
6. Identify the odd-Mii out.

Task 4: Fastest Mii

In this task, we used our custom classifier (i.e. feature vectors were generated using: **Histogram of Oriented Gradients and Boosted Decision Trees** was used for classification) to recognize faces. Once the classifier output was generated, we tried doing **non-maxima suppression** but that did not give us very accurate face detection as outliers were still present. So we employed the **mean shift clustering algorithm** to group detected faces together and cluster them. Once this was done, we obtained the cluster centers which represented the center of the rectangles bounding the faces. Using these faces as **Region of Interests** in the **Shi-Tomasi algorithm**, we obtained features to track. Here we employed another new technique, we (1) eliminated high error cases (using the magnitude of the flow vector), (2) eliminated errors (using the angles between the flow vectors – as most vectors in the region need to be parallel). After this, we used the average of the remaining vectors as the flow vector for the region. (**Improvisation:** Normal average of all vectors is not a good representative, so we took the **average of the medians of groups of vectors**.) This process was repeated over 10 frames and the maximum magnitude vector was taken as the flow of the fastest mii. As 10 frames might not be sufficient to get the fastest mii, we ran this process 4 times (i.e. overall 10*4 = 40 frames were considered) and output the **maximum among these 4 iterations as the fastest mii**.