Problem Definition

- Given a video and a fixed object class, track the motion of all objects of the given class over time.
- Very important problem with many applications:
- Surveillance.
- Traffic control.
- Augmented reality.
- Medical imaging.
- Robotics.
- Video editing.
- What makes the problem hard.
- Multiple objects.
- Motion can be complex.
- Objects can be nonrigid.
- Partial or full occlusions.
- Entries and exits can occur.
- Complex object shapes.
- Scene illumination changes.
- Real-time processing requirement.

Salient Features

- One of the most important parts of object tracking.
- Color.
- Perceptually uniform color spaces with uncorrelated components (Luv, Lab, HSV, etc.) are better.
- Histograms. Comparing histograms.
- Edges.
- Many popular methods. Canny edge detector.
- Motion.

- Computed used optical flow methods.
- Texture.
- Haar wavelets. Invariant to background noise, used in many methods.

Types of Object Tracking

- Point tracking. Each object is associated with a point, and we must match up points between consecutive frames.
- Discrete: minimize correspondance cost.
- Stochastic: Kalman filters, particle filters. Based on discrete-time dynamical system to estimate the state given all measurements up to that point. These tend not to work well because of overly restrictive assumptions on tracjectory of object or on distribution of noise. Once the tracker loses track of the object, it is usually gone for good.
- Kernel tracking. Each object is associated with a kernel (simple geometric shape), and features computed within the kernel are used to estimate motion from one frame to the next.
- Silhouette tracking.
- Contour evolution.
- Matching shapes – similar to template matching. Object is only assumed to translate from frame to frame.

Unsupervised Approaches

- Background Subtraction
- Build a representation of the scene called a background model, and find deviations from the model for each incoming frame.
- Connected components algorithm used to find closed regions corresponding to foreground objects.
- Main types of approaches:
  - Modeling per-pixel background distribution, e.g. by fitting a mixture of Gaussians to each pixel.
  - Supposedly achieves better results for outdoor scenes involving periodic (swaying of trees, etc.) motion and shadows.
  - Use HMM to associate each pixel with a state.
- Mean shift.
- Requires lots of manual tuning to obtain acceptable results.
- Show using picture.
- Graph cuts.
- Requires fewer manually-selected parameters than mean shift.
- Expensive both in terms of space and time complexity.

- Active contours. Start with an initial estimate of contour around an object, and shrink the contour over time such that a prescribed energy functional is minimized.
- Not discussed here.

Deficiencies of Previous Methods

- Advantages of previous two methods:
- Candidate object regions are easily found.
- Candidate regions found without any features specifically designed to discriminate objects of the given class. This is the reason why they fail.
- Disadvantages:
- Requires prior access to video sequence of the same scene with the same camera orientation so that we we can extract the background in the preprocessing step.
- What pictures show:

(a) If an object stays still for a large portion of the video sequence, then it will be considered part of the background.
(b) Background subtraction is sensitive to noise. Candidate regions can spontaneously pop up at random locations of the video due to noise.
(c) Even the methods that were supposedly designed to be invariant to shadows do not work. Background subtraction is only reliable when the camera orientation and illumination conditions are such that shadows do not affect the object contours.
(d) If colors of object do not deviate much from mixture model, then the object will be considered part of the background.
(e) and (d) A single object is often segmented into multiple disconnected components. Figuring out how to merge the components so that most of them contain only one object requires a lot of manual tuning dependent on the scene constraints (illumination, motion, etc.).

What Works

- Mention benchmark paper and what it does.
- Robustness evaluation measurements:
  - One-pass robustness evaluation (OPE). Initialize from ground truth at frame 1.
  - Temporal robustness evaluation (TRE). Initialize from ground truth at varied starting frame.
  - Spatial robustness evaluation (SRE). Initialize from ground truth with bounding box offset from ground truth.
- Criteria for evaluating algorithms:
  - Illumination variation.
  - Scale variation.

  - Occlusion.
  - Deformation.
  - Motion blur.
  - Fast motion.
  - In-plane rotation.
  - Out-of-plane rotation.
  - Out-of-view.
  - Background clutters.
  - Low resolution.

- All of them are kernel trackers.
- Both use training phase to learn features from image sequences.
- Two main approaches to classification and tracking:
- Given the position of an object at frame t, sample nearby regions in frame $t + 1$, and choose the patch that maximizes a similarity metric.
  - Problems with this approach.
  - Objective of classifier does not necessarily coincide with objective of tracker.
  - The assumption that the patch achieving the best classifier confidence corresponds to the best estimate of the position of the object in the next frame does not always hold.
    * Example: view changes or similar-looking nearby objects.
- Perform the object classification and tracking jointly by predicting a position vector.
- Online learning.
- The features need to be updated over time to account for variation in object appearance and background.
  - Which features are good? The bounding box surrounding the tracked object will contain background pixels. How can we make the features invariant to this?
  - Illumination conditions, non-rigid objects, etc.
- How frequently to update training examples?
  - Too infrequent means that the tracker will be susceptible to changes in object appearance and background variation.
- How to generate new training examples?
  - How to choose candidate training examples, and how to determine labels?
  - If we choose the labels incorrectly, then the tracker may drift over time and eventually lose track of the target.
  - How to associate training examples with measure of confidence as to how confidently they should be classified positive or negative?
- How to choose which old features to update?
  - The old features initialized during the start tend to be good for a long time.

  - If we update too many features, the tracker will tend to drift and lose track of the target object.
  - How to update features to handle occlusions well?

SCM and ASLA

- Sparse active appearance models.
- Given the position of the object at frame t, both methods sample and evaluate several candidate patches in frame $t+1$ to determine the best match.
- During training, both methods extract patches nearby the target region, and learn a sparse projection matrix S.
- This matrix projects the a given patch or set of patches to a lower-dimensional feature space.
- The matrix S is obtained by solving an ℓ_1 regularized least-squares problem during training.
- ℓ_1 regularization is important because it allows us to learn a small number of salient features that will not change much over time. This keeps the tracker stable.
- After this point, both methods differ. But they both exploit sparsity of the features used in the active appearance models.
- SCM uses the the sparse reprojection matrix as a discriminative classifier, and also trains a generative classifier.
- Each classifier is used to generate a confidence score for each candidate patch in frame $t+1$.
- The score generated by the discriminative classifier is based on the reprojection error confidence.
- The generative classifier uses ℓ_1 regularized least squares to learn a dictionary of k-means clusters.
- The coefficient vectors that minimize the reprojection error for each patch are concatenated to form a histogram that is modified to take occlusion into account.
- A confidence score is generated using a histogram similarity measure between the object and each candidate patch in frame $t+1$.
- The candidate region selected in frame $t+1$ is chosen such that the product of the reprojection confidence using the discriminate classifier and the histogram similarity confidence using sparse coding.
- ASLA forms a sparse matrix from the coefficient vectors extracted from sparse reconstruction using alignment pooling.

- To perform the template update, it associates each template with an update probability.
- Older templates are associated with small update probabilities, and newer templates are associated with large update probabilities.
- To choose which template to update, we generate a random number in the interval $[0, 1]$, and choose a template to update based on the section we land in.
- Tracking is done using a Bayesian inference framework, where we assume that the motion between frames is affine.
- Difference in how they characterize background.
- SCM models both positive (foreground) and negative (background) patches in the sparse projection step.
- ASLA only uses templates within the bounding box for the object.
- SCM tries to model the object uses both foreground and background information. ASLA tries to model the object using information within the object boundary itself.

Struck

- Traditional discriminative classification approaches aim to learn a similarity measure between two object regions.
- Adaptation is separated into two parts:
- Generation and labeling of samples.
- Updating of classifier.
- Issues with this approach:
- Not clear how to generate and label training samples in principled manner. Usual approaches rely on predefined rules, which do not always work well.
- Objective of the classifier (predicting binary label correctly) does not always coincide with objective of tracker (estimate object location correctly).
- The assumption that the best classifier confidence corresponds to the best estimate of the object does not always hold.
- State of the art methods focus on improving the robustness of the classifier to poorly-labeled samples.
  - Recall that one of the main problems is obtaining representative samples.
- Struck uses a *structured output SVM* that models a discriminative function $F(x_t, y_t)$, where x_t is an object region and y_t is the translation from x_t to the new position in frame $t + 1$.

- Support vector machines tend to do better than boosted online classifiers because:
- They are robust to label noise. Recall that AdaBoost in particular is very sensitive to noise.
- This approach makes use of structured output SVMs.
- This approach uses the same architecture to perform the classification and tracking.
- Online learning with kernels suffers from the fact that the number of support vectors increases with the amount of training data.
- Compare this to the template updating in the other methods.
- Constrain number of support vectors to be under some fixed limit.
- Modify the constraints of the standard binary classification SVM to ensure that for a given training sample (x_i, y_i), the transformation y_i achieves the maximum compatibility $F(x, y_i)$ defined by $\langle x, \phi(x, y) \rangle$ (the output of the SVM) by a margin determined by a loss function.
- The loss function can be the bounding box overlap, for example.
- As a result of the formulation, there are only two kinds of support vectors for a given *support pattern* x_i:
- Only the support vector (x_i, y_i) has $\beta_i^{y_i} > 0$.
- Other support vectors with $y \neq y_i$ have $\beta_i^{y_i} < 0$.
- We refer to these as positive and negative support vectors, respectively.
- Optimization is performed using a modified version of the SMO algorithm. While updating the gradient during SMO, we have three options that map well into the object tracking framework:
- Process a new example (x_i, y_i), add the true label y_i as a positive support vector, and search for the most important sample to use as a negative support vector.
- Process an old example x_i, modify the associated negative support vector, and adjust coefficients.
- Process an existing support pattern x_i at random, but only modify existing support vectors.
- Only options 1 and 2 can add support vectors.
- When searching for negative background examples, we sample in a polar grid instead of using a dense pixel-by-pixel search. Note that this is what may have caused the decreased location and overlap precision compared to other methods.
- Incorporating a budget:

- Remove the support vector that results in the smallest change $\|w\|^2$ to the weight vector w.
- It is sufficient to consider the removal of only the negative support vectors.
- In case a support pattern has only two support vectors, both must be removed.
- Does not significantly affect tracking performance.
- Kernel functions:
- We crop a patch around the region in the next frame corresponding to the candidate motion vector, and apply a standard image kernel to features extracted from both patches.
  - E.g. Gaussian kernel.
- Traditional discriminative classification approaches aim to learn a similarity measure between two object regions.
- Adaptation is separated into two parts:
- Generation and labeling of samples.
- Updating of classifier.
- Issues with this approach:
- Not clear how to generate and label training samples in principled manner. Usual approaches rely on predefined rules, which do not always work well.
- Objective of the classifier (predicting binary label correctly) does not always coincide with objective of tracker (estimate object location correctly).
- The assumption that the best classifier confidence corresponds to the best estimate of the object does not always hold.
- State of the art methods focus on improving the robustness of the classifier to poorly-labeled samples.
  - Recall that one of the main problems is obtaining representative samples.
- Struck uses a *structured output SVM* that models a discriminative function $F(x_t, y_t)$, where x_t is an object region and y_t is the translation from x_t to the new position in frame $t + 1$.
- Support vector machines tend to do better than boosted online classifiers because:
- They are robust to label noise. Recall that AdaBoost in particular is very sensitive to noise.
- This approach makes use of structured output SVMs.

- This approach uses the same architecture to perform the classification and tracking.
- Online learning with kernels suffers from the fact that the number of support vectors increases with the amount of training data.
- Compare this to the template updating in the other methods.
- Constrain number of support vectors to be under some fixed limit.
- Modify the constraints of the standard binary classification SVM to ensure that for a given training sample (x_i, y_i), the transformation y_i achieves the maximum compatibility $F(x, y_i)$ defined by $\langle x, \phi(x, y) \rangle$ (the output of the SVM) by a margin determined by a loss function.
- The loss function can be the bounding box overlap, for example.
- As a result of the formulation, there are only two kinds of support vectors for a given *support pattern* x_i:
- Only the support vector (x_i, y_i) has $\beta_i^{y_i} > 0$.
- Other support vectors with $y \neq y_i$ have $\beta_i^{y_i} < 0$.
- We refer to these as positive and negative support vectors, respectively.
- Optimization is performed using a modified version of the SMO algorithm. While updating the gradient during SMO, we have three options that map well into the object tracking framework:
- Process a new example (x_i, y_i), add the true label y_i as a positive support vector, and search for the most important sample to use as a negative support vector.
- Process an old example x_i, modify the associated negative support vector, and adjust coefficients.
- Process an existing support pattern x_i at random, but only modify existing support vectors.
- Only options 1 and 2 can add support vectors.
- When searching for negative background examples, we sample in a polar grid instead of using a dense pixel-by-pixel search. Note that this is what may have caused the decreased location and overlap precision compared to other methods.
- Incorporating a budget:
- Remove the support vector that results in the smallest change $\|w\|^2$ to the weight vector w.
- It is sufficient to consider the removal of only the negative support vectors.
- In case a support pattern has only two support vectors, both must be removed.

- Does not significantly affect tracking performance.
- Kernel functions:
- We crop a patch around the region in the next frame corresponding to the candidate motion vector, and apply a standard image kernel to features extracted from both patches.
  - E.g. Gaussian kernel.

Comparison

- SCM and ASLA have Best overall performance if accurate tracking is required (large overlap threshold with ground truth bounding box, and small permissible location error).
- Their success indicates that sparse representations are effective models to account for appearance change, e.g. occlusion.
- Local sparse representations seem to be more effective the holistic sparse representation templates.
- The alignment pooling technique of ASLA is robust to misalignments and background clutter.
- SCM and ASLA both perform horribly in sequences where there is fast motion, because both only sample locally.
- Struct has best overall performance if only very rough tracking is required (small overlap threshold with ground truth bounding box, and large permissible location error). This is because Struct only estimates location and does not handle scale variation.
- Does not handle scale large changes during video as well as SCM and ASLA.
- Does not handle occlusion as well as SCM and ASLA.

Conclusion

- Background subtraction and image segmentation do not work well because they do not incorporate discriminative features of the object that is being tracked. They also do not adapt to account for appearance and background changes.
- SCM and ASLA use sparse active appearance models and local sampling. They do well when the video does not contain fast motion.
- Struck uses structured output SVMs to perform the classification and tracking jointly. It does well when the scale of the object does not change significantly during the video.
- An approach that combines ideas form both of these approaches to address the weaknesses of each would be desirable.