Interactive Feature Tracking using K-D Trees and Dynamic Programming

A faster than real-time tracker:
- with a human in the loop, which is modelled by the system,
- a comprehensive track definition allowing optimal interactive tracking with massive automatic preprocessing.

Extremely simple and intuitive user-guided process:
1. Single user click generates a track through the entire sequence at over 100fps.
2. Further clicks lock down keyframe features and improve the track.

Prior State of the Art:
- including Mean-shift, KLT, WSL, etc,
- do not model the possibility of failure,
- fall even on ‘easy’ sequences, due to
  - “Track-to-first”—do not model appearance change,
  - “Track-to-previous”—track is lost through drift,
- require user to continually monitor and restart,
- time taken to track sequence: >100% of sequence length.

Instead, User-in-Loop Batch Tracking:
- user specifies keyframe features, one at a time,
- the optimal track over the entire sequence is generated,
- optimizing over
  - smoothness of trajectory,
  - similarity to keyframe features,
- smoothness of appearance,
- summed occlusion penalties,
- tuning parameters can be updated interactively,
- time taken to track sequence: 40% of sequence length.

Our approach overcomes traditional conflict between:
1. Small search region—tracking is fast per frame; must use every frame; unable to recover after occlusions.
2. Large search region—tracking is slow per frame, but subsampling allows speed-ups; recovery possible after occlusions, but bad tracks due to mismatches.

Input: Image stream, \( I(x, y, t) \); keyframe features, \( Q = \{ (x_i, y_i, t_i, q_i) \} \); tuning parameters, \( \mu_{1,2} \).

Output: Optimal track, \( T = \{ (x_i, y_i, p_i) \} \), minimizing:
\[
E(T) = \sum_{i} \mu_1 \|x_i - x_{i-1}\|^2 + \mu_2 \|p_i - p_{i-1}\|^2 + \min \|q_i - q\|^2
\]
*note: norms are robust for occlusions and multichannel*

with \( p_i \) being the vectorized appearance of the feature at \( x_i \) in frame \( t \) and \( q_i \) the appearance of the \( i^{th} \) keyframe feature.

Immense speed-ups, taking this optimization to interactive rates, are achieved by separating detection from tracking:
- **Fast detection** is facilitated by converting each image to a 16D point cloud, using convolution with a filter bank (obtained via PCA of natural image patches), and storing it in a K-D tree.
- The **optimal track** is efficiently extracted using dynamic programming (Viterbi algorithm) from the small set of candidate matches for each frame selected by the detection stage.

Occlusions are incorporated by treating them as a special type of feature.

Refinement automatically updates the track trajectory to **sub-pixel precision**. This is required because the image locations of the data points in the K-D trees are necessarily quantized.