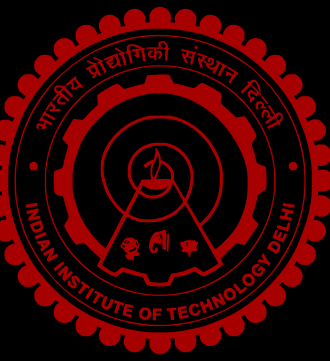
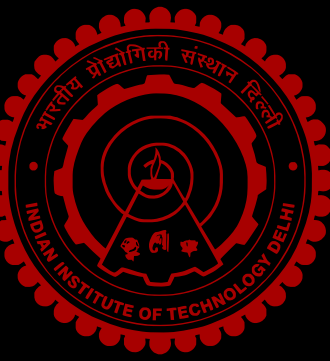




Feature-based Template Tracking

1. Detect feature and *produce key points* (with descriptor vectors) in both frames
2. For each feature *find its matching feature* in the target (nearest L_2^2 ?)
 - Approximate nearest neighbor search using a high-dimensional search tree (see FLANN, *e.g.*)
 - Actually find the nearest two, and only accept the top match if it is significantly closer than the second best (Lowe's Ratio Test)
 - If feature A_1 of frame F_1 is the closest match (in F_1) for feature A_0 of frame F_0 , then A_0 should also be the closest match for A_1 .. Otherwise, *discard the match*
3. *Fit a plane* to features (using RANSAC for outlier removal)
4. *Find the homography* between the two planes
 - Use it to warp the initialized region into the new frame

Fundamental Matrix for general tracking
i.e., features are not on plane



Tracking in Practice

- **Block-based methods:** fast, but less robust *e.g., detect UI icons, fixed labels, fixed objects in assembly lines*
 - OK for rigid, flat objects not near camera, with constant lighting, and panning camera
- **Differential methods:** efficient for smooth moderate motion *e.g., ego-motion stabilization, visual odometry, optical mice, hand tracking*
 - Can provide sub-pixel accuracy, but accumulates drift and can lose objects
- **Feature-based trackers** are the most robust, but also the slowest (**efficient with markers**)
 - Immune to scale & rotation, tracking drift, occlusion, but require textured, complex objects, and bounding boxes can appear jittery due to change of selected features

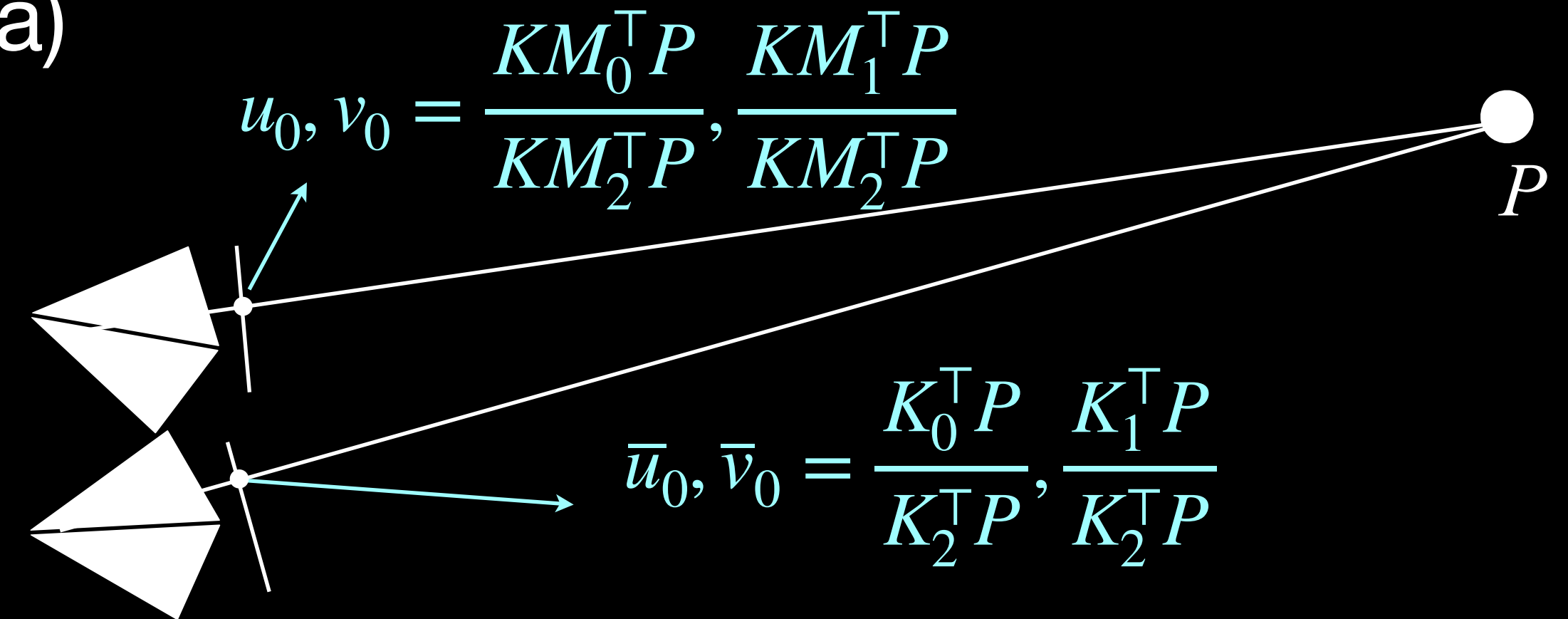
Use hybrid methods, including ML-based methods
Use Kalman filter to combine methods

e.g., spatial tracking in AR, drone localization (SLAM), and tracking warping targets

Deriving 3D Structure

Multiple images of the same scene

- Structure from motion (of calibrated camera)
 - Track to find many corresponding pairs: p, q
 - Find essential matrix E using all such pairs
 - Has 7 DOF, one less than homography
 - Factorize E to get relative camera pose, M
 - Triangulate to obtain the 3D point for each pair (p, q)
 - Refine using bundle adjustment
- Stereo-image reconstruction is similar



$$(KM_0^T - u_0 KM_2^T)P = 0$$

$$(KM_1^T - v_0 KM_2^T)P = 0$$

$$(K_0^T - \bar{u}_0 K_2^T)P = 0$$

$$(K_1^T - \bar{v}_0 K_2^T)P = 0$$

Solve for P

Epipolar Constraint

- A match for p (in frame1) must lie on the corresponding **Epipolar line** l_p (in frame2)
- The **epipolar line** is the projection of the infinite 3D ray $\pi^{-1}(p)$ in the other camera

Epipolar plane: $Cam, \overline{Cam}, p$
 $\pi^{-1}(p)$ lies on the Epipolar plane

Fundamental matrix F :

$$l_p = Fp \text{ and } l_{\bar{p}} = F^T \bar{p}$$

$$\bar{p}^T F p = 0 \text{ and } p^T F^T \bar{p} = 0$$

Essential matrix $E = \bar{K}^T F K$

maps points given in
normalized image planes

