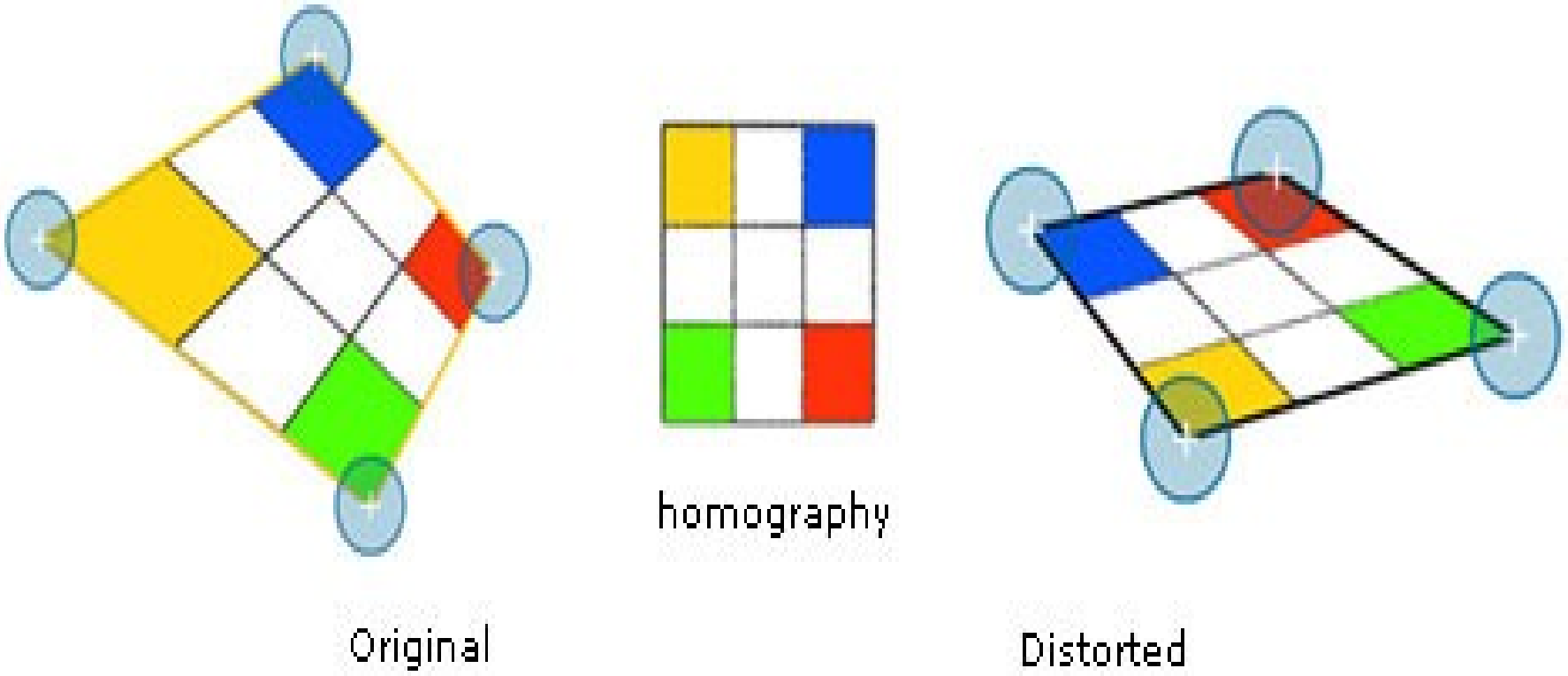
Planar tracking for 3d reconstruction

Planar tracking for 3d reconstruction

- Context : 3d video editing
- No occlusion
- Planar surface (only homography transformations)
- Need nearly perfect matching
  => errors amplified in 3d reconstruction
- User interaction (to avoid occlusion in the surface, to correct if an error occurs)

Homography tracking



http://en.nicoptere.net/?p=59

Homography tracking

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} H_{00} & H_{01} & H_{02} \\ H_{10} & H_{11} & H_{12} \\ H_{20} & H_{21} & H_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} \Leftrightarrow \mathbf{p}_2 = H\mathbf{p}_1$$

- Z1 = 1
- Screen coordinates : (X2/Z2, Y2/Z2)

Homography tracking

Can be computed with at least 4 corresponding points :
(set H_{00} = a, H_{01} = b, …, H_{21} = h, H_{22} = 1)

$$\begin{pmatrix} u_0 & v_0 & 1 & 0 & 0 & 0 & -u_0x_0 & -v_0x_0 \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3x_3 & -v_3x_3 \\ 0 & 0 & 0 & u_0 & v_0 & 1 & -u_0y_0 & -v_0y_0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1y_1 & -v_1y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2y_2 & -v_2y_2 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -u_3y_3 & -v_3y_3 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{pmatrix} = \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

Features approach

- Very few features if the planar area if small (often the case in video editing context)
  => worse features

Template matching approach

- Patch comparison : SSD (or other score function)
- Full homography tracking
  => 8 degrees of freedom
- Iterative homography tracking :
  Starts by optimizing translation (H_{02} and H_{12} only)
  Then add rotation (H_{00}, H_{01}, H_{10}, H_{11})
  Then shear (H_{20}, H_{21})

Iterative homography tracking

- Problem :
  difficult to optimize the 4 rotations params at the same time
- Solution :
  start by optimizing θ such that :
  $H_{00} = \cos(\theta)$, $H_{01} = -\sin(\theta)$, $H_{10} = \sin(\theta)$, $H_{11} = \cos(\theta)$
  Then optimize on the 4 parameters, it should be nearer to the solution.

Iterative homography tracking

- On my implementation :
  optimisation limited to translation and θ, adding more takes a lot of time and doesn't give better results.

  gives good tracking on a few test (no benchmark for now), but a few pixels decal on the borders of the patch.

  the homography matrix isn't correct (5 degrees of freedom are missing).

  => Local tracking

Local tracking

- First tracking => position of the patch nearly known
- Improve by tracking local features with very small search window (tracking by SSD).
- Avoid too much features at the same place
- Recompute full homography matrix based on local tracking.

Local tracking

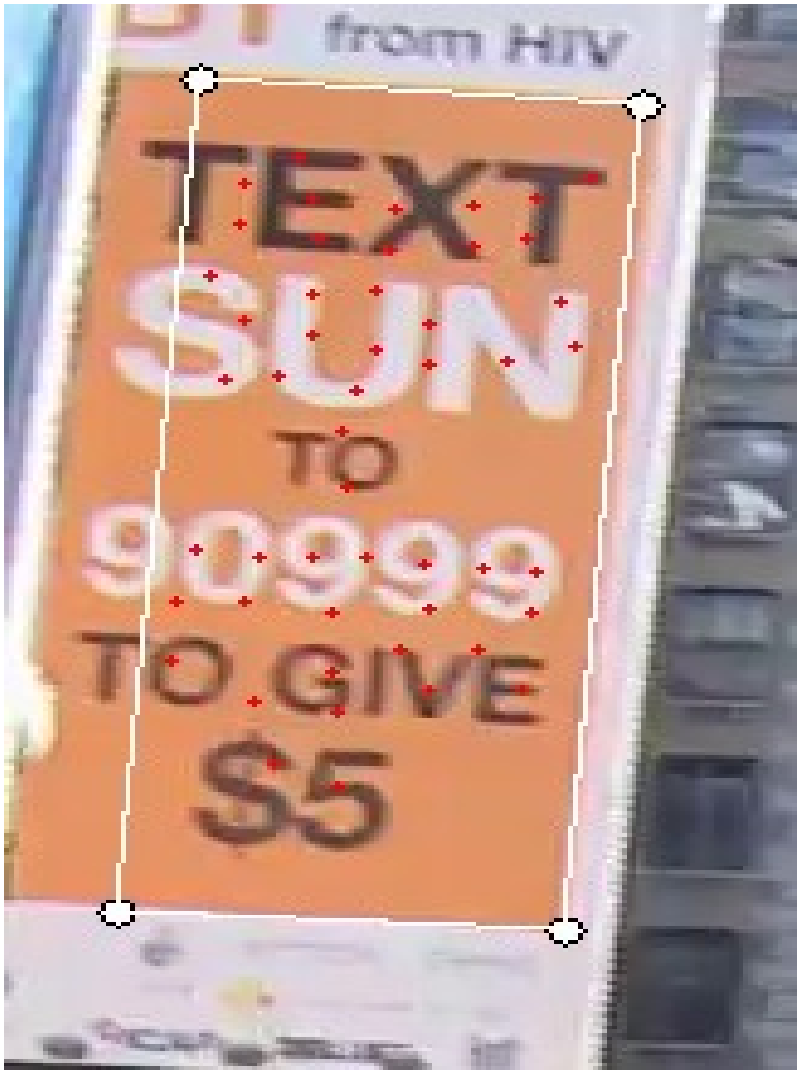
- How to choose the features?

  For each pixel from the initial patch, apply a pattern matching on itself.

  Keep pixels with the best distance maps (small distance on the center, high values on the borders).

  Integral map can be use to reduce the complexity of testing each position (if pixel by pixel comparison used).

Local tracking



Result

- The tracking is improved (no benchmark for now)

Project interface