動画像を用いた高度道路交通システム

Jien kato
Virtual View Generating Using Image-based Rendering (IBR)
Objective

Generating image stream of a virtual view point using images from multiple static cameras

Applications
- ITS (safe driving assistance)
- Security surveillance
- Entertainment
Related work (1/3)

Feature detection

- Detect features using SIFT [Lowe, IJCV 2004]
Related work (2/3)

Pairwise feature matching
- match features between each pair of images
Related work (3/3)

Refine matching using RANSAC [Fischler & Bolles 1987]
Approach

Our approach extends view morphing (a IBR method based on implicit geometry) by integrating:
- Robust fundamental matrix estimation
- Feature matching

View morphing has not been wildly used in real application:
- Shape preserved
- Disadvantage
  - Excess needs of manual operations
  - Need of a prior knowledge of scene geometry
Parallel view: situation where linear interpolation of images is shape-preserving

\[ \Pi_1 = \begin{bmatrix} f_1 & 0 & 0 & -f_1 C_X \\ 0 & f_1 & 0 & -f_1 C_Y \\ 0 & 0 & f_1 & 0 \end{bmatrix} \]

\[ \Pi_0 = \begin{bmatrix} f_0 & 0 & 0 & 0 \\ 0 & f_0 & 0 & 0 \end{bmatrix} \]

Linear interpolation:
\[ (1-s)p_0 + sp_1 = (1-s)\frac{1}{Z} \Pi_0 P + s\frac{1}{Z} \Pi_1 P \]
\[ = \frac{1}{Z} \Pi_s P \]

New perspective view:
\[ \Pi_s = (1-s)\Pi_0 + s\Pi_1 \]
\[ C_s = (sC_X, sC_Y, 0) \]
\[ f_s = (1-s)f_0 + sf_1 \]
View morphing

Three steps

1. **Prewarping**
   - Original images \(I_0, I_1\) are prewarped to form parallel views \(\hat{I}_0, \hat{I}_1\).

2. **Morph**
   - \(\hat{I}_s\) is produced by interpolating the prewarped images.

3. **Postwarping**
   - \(\hat{I}_s\) is postwarped to form \(I_s\).
Prewarping (1/4)

Projective transforms: $H_0, H_1$
- Each consisting of rotations

$R^d_i$: rotation of angle $\theta_i$ about axis $d_i$ in depth which makes the image plane become parallel

$R^\phi_i$: Affine warping to align the scanlines
Prewarping (2/4)

How we do prewarping?

1. Estimate fundamental matrix: F
   - Using 8-point algorithm
     - Select a set of key points from $I_0, I_1$ by SIFT
     - Match key points between image pair
       - Finding the nearest neighbor of their descriptors in Euclidean distance
     - Estimate F via RANSAC iterations
       - Linear estimation from 8 points correspondence
Prewarping (3/4)

② Factorize $F$ with singular value decomposition

- Obtain two unit eigenvectors

$$e_0 = [e_0^y, e_0^x, 0], \quad e_i = [e_i^y, e_i^x, 0]$$

- Select rotation axes as

$$d_0 = [- e_0^y, e_0^x, 0], \quad [x, y, z] = F d_0, \quad d_1 = [- y, x, 0]$$

- Calculate angles

$$\theta_i = - \frac{\pi}{2} - \tan^{-1}\left( \frac{d_i^y e_i^x - d_i^x e_i^y}{e_i^z} \right)$$

$$\phi_i = - \tan^{-1}\left( \frac{\tilde{e}_i^y}{\tilde{e}_i^x} \right)$$
Prewarping (4/4)

③ Do projective transform

- After image been rotated twice,
  \[
  \tilde{F} = R_{\phi_1} R_{\theta_1}^d F R_{-\theta_0}^d R_{-\phi_0} = \begin{bmatrix}
  0 & 0 & 0 \\
  0 & 0 & a \\
  0 & b & c 
\end{bmatrix}
  \]

- F should be in the form
  \[
  (H_1^{-1})^T F H_0^{-1} = \begin{bmatrix}
  0 & 0 & 0 \\
  0 & 0 & -1 \\
  0 & 1 & 0 
\end{bmatrix}
  \]

- Introduce a translation T on \( I_1 \)

- Calculate projective transform
  \[
  H_0 = R_{\phi_0} R_{\theta_0}^d, H_1 = TR_{\phi_1} R_{\theta_1}^d
  \]

\[ T = \begin{bmatrix}
  0 & 0 & 0 \\
  0 & -a & -c \\
  0 & 0 & b 
\end{bmatrix} \]
Image Interpolation (1/2)

Feature correspondence
- Need a sufficient number of matches between source images and they should be equally distributed.
  - Difficult in case of images with wild baseline

How we do interpolating
① Jointly use SIFT and Harris detectors
② Corner point: normalized cross-correlation
   SIFT point: nearest neighbor in descriptor
Image Interpolation (2/2)

How we do interpolating

③ Eliminate the false outliers
   - Use the precomputed fundamental matrix to remove outliers by enforcing Epipolar constrain
     \[ \tilde{m}'^T \tilde{F} \tilde{m} = 0 \]
     - Obtain a set of sufficient correspondence with high confidence
   - If necessary, improve the results by hand.

④ Do image warp and cross-dissolve
Experiments (1/6)

Apply our method to intersection images
- Generate a reference view that dynamically follows host vehicle at high position

Camera setting
- Six cameras, not calibrated in advance
- Each one and clockwise neighbor form a pair $C_{n_0}, C_{n_1}$
- Onboard system is supposed to receive the image stream while approaching intersection
Experiments (2/6)

Assume online direction of host vehicle is known as $\omega_t$

Choose camera pair $n$ which has the closest direction with $\omega_t$

Approximate morphing rate $s$ by

$$s_t = \frac{\omega_{t0}}{\omega_{t0} + \omega_{t1}}$$

$$\gamma = (\omega_{t1} - \omega_{t0})$$
Experiments (3/6)

Source image streams

Distance of viewpoints is around 60°
Experiments (4/6)

对其进行预失真

before

after
Experiments (5/6)

Results of SIFT & Harris-Corner detectors

+ : SIFT
+ : Harrs
Experiments (6/6)

An image stream from a virtual viewpoint
Conclusion

To put view morphing into practice use, it is necessary to reduce manual operations.

Our approach has shown a solution:
- It is capable to work on images even taken from disparate view points:
  - Without any prior knowledge of scene geometry
  - Without excess manual operation
- Results are good.

Need more tests.