Inside ARToolKit

Hirokazu Kato
Hiroshima City University
kato@sys.im.hiroshima-cu.ac.jp
http://www.sys.im.hiroshima-cu.ac.jp/people/kato/
But

- 40 min. is too short to talk everything.
- Focus into some important points.
- This is for people who has developed applications with ARToolKit.
Outline

1. Mathematical & Algorithm Background
   • Pose & Position Estimation
   • Rectangle Extraction

2. Implementation
   • Camera Calibration
   • Image Processing
   • Pose Estimation
   • Background Video Stream Display
1.1 Pose & Position Estimation

- Coordinates System
- Equations
- Calculation
- Initial Condition Problem
Coordinate Systems
Relationships: Marker & Camera

Rotation & Translation

\[
\begin{bmatrix}
X_C \\
Y_C \\
Z_C \\
1
\end{bmatrix} =
\begin{bmatrix}
R_{11} & R_{12} & R_{13} & T_1 \\
R_{21} & R_{22} & R_{23} & T_2 \\
R_{31} & R_{32} & R_{33} & T_3 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
X_M \\
Y_M \\
Z_M \\
1
\end{bmatrix} = T_{CM}
\]

\[
\begin{bmatrix}
X_M \\
Y_M \\
Z_M \\
1
\end{bmatrix}
\]
Relationships: Camera & Ideal Screen

Perspective Projection

\[
\begin{bmatrix}
hX_l \\
hY_l \\
h
\end{bmatrix} = \begin{bmatrix}
sf_x & 0 & x_c & 0 \\
0 & sf_y & y_c & 0 \\
0 & 0 & 1 & 0
\end{bmatrix} \begin{bmatrix}
X_c \\
Y_c \\
Z_c \\
1
\end{bmatrix} = C
\begin{bmatrix}
X_c \\
Y_c \\
Z_c \\
1
\end{bmatrix}
\]

C : Camera Parameter
Image Distortion
Image Distortion Parameters

Relationships between Ideal and Observed Screen Coordinates

\[ d^2 = (x_i - x_0)^2 + (y_i - y_0)^2 \]
\[ p = \{1 - fd^2\} \]
\[ x_o = p(x_i - x_0) + x_0, \quad y_o = p(y_i - y_0) + y_0 \]

\((x_0, y_0)\) : Center Coordinates of Distortion
\(f\) : Distortion Factor
Scaling Parameter for Size Adjustment

- Ideal Image → Distorted Image
- Observed Image → Compensated Image (doesn't fit the screen) → Scale adjusted Image
Implementation of Image Distortion parameters

\[ x = s(x_i - x_0), \quad y = s(y_i - y_0) \]
\[ d^2 = x^2 + y^2 \]
\[ p = \{1 - fd^2\} \]
\[ x_d = px + x_0, \quad y_d = py + y_0 \]
\[ \text{dist} \_ \text{factor}[0] = x_0 \]
\[ \text{dist} \_ \text{factor}[1] = y_0 \]
\[ \text{dist} \_ \text{factor}[2] = 1000000000.0 * f \]
\[ \text{dist} \_ \text{factor}[3] = s \]
What is pose & position estimation?

Marker Coordinates: \((X_m, Y_m, Z_m)\)

Camera Coordinates

Ideal Screen Coordinates

Observed Screen Coordinates: \((x_0, y_0)\)

\(T_{mc}: \text{known}\)

\(X_m, Y_m, Z_m, x_0, y_0\) are known

GET by Image Processing
How to get $T_{CM}$
Search $T_{cm}$ by Minimizing Error

Optimization

- Iterative process

\[
\begin{bmatrix}
    h\hat{x}_i \\
    h\hat{y}_i \\
    h
\end{bmatrix} = C \cdot T_{cm} \begin{bmatrix}
    X_{Mi} \\
    Y_{Mi} \\
    Z_{Mi} \\
    1
\end{bmatrix}, \quad i = 1,2,3,4
\]

\[
err = \frac{1}{4} \sum_{i=1,2,3,4} \left\{ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right\}
\]
How to set the initial condition for optimization process

Geometrical calculation based on 4 vertices coordinates
- Independent in each image frame: Good feature.
- Unstable result (Jitter occurs.): Bad feature.

Use of information from previous image frame
- Needs previous frame information.
- Cannot use for the first frame.
- Stable results. (This does not mean accurate results.)

ARToolKit supports both.
See 'simpleTest2'.
1.2 Rectangle Extraction

1. Thresholding, Labeling, Feature Extraction (area, position)
2. Contour Extraction
3. Four straight lines fitting
   • Little fitting error => Rectangle.

This method is very simple. Then it works very fast.
2. Implementation

- Camera Calibration Method
- Image Processing
- Pose Estimation
- Background Video Stream Display
2.1 Camera Calibration

- Camera Parameters
  1. Perspective Projection Matrix
  2. Image Distortion Parameters

- ARToolKit has 2 methods for camera calibration.
  1. Accurate 2 steps method
  2. Easy 1 step method
Accurate 2 steps method

Using dot pattern and grid pattern

2 step method

• 1) Getting distortion parameters
• 2) Getting perspective projection parameters
Step 1: Getting distortion parameters 'calib_dist'

Selecting dots with mouse

- Take pattern pictures as large as possible.
- Slant in various directions with big angle.
- 4 times or more

Getting distortion parameters by automatic line-fitting
Step 2: Getting perspective projection matrix 'calib_cparam'

Manual line-fitting

Grid cardboard have to be moved in the perpendicular direction of the plane. Camera should be placed in almost perpendicular direction of the plane.
Easy 1 step method: 'calib_camera2'

- Same operation as 'calib_dist'.
- Getting all camera parameters including distortion parameters and perspective projection matrix.
- Not require careful setup.
- Accuracy is good enough for image overlay. (Not good enough for 3D measurement.)
Camera Parameter Implementation

**Camera parameter structure**

- `typedef struct {
    int         xsize, ysize;
    double      mat[3][4];
    double      dist_factor[4];
} ARParam;

**Adjust camera parameter for the input image size**

- `int arParamChangeSize( ARParam *source, int xsize, int ysize, ARParam *newparam );`

**Read camera parameters from the file**

- `int arParamLoad( char *filename, int num, ARParam *param, ... );`
2.2 Notes on Image Processing

- Image size for marker detection
- Use of tracking history
- Accuracy v.s. Speed on pattern identification
(1) Image size for image processing

**AR_IMAGE_PROC_IN_FULL**
- Full size images are used for marker detection.
- Taking more time but accuracy is better.
- **Not good for interlaced images**

**AR_IMAGE_PROC_IN_HALF**
- Re-sampled half size images are used for marker detection.
- Taking less time but accuracy is worse.
- Good for interlaced images

**External variable**: arImageProcMode in 'ar.h'
**Default value**: DEFAULT_IMAGE_PROC_MODE in 'config.h'
(2) Use of tracking history

Marker Detection
• With tracking history: arDetectMarker();
• Without tracking history: arDetectMarkerLite();

How to use tracking history
• Error correction of pattern identification
• Lost marker insertion
(3) Accuracy v.s. Speed on pattern identification

Pattern normalization takes much time. This is a problem in use of many markers.

- Normalization process.

Normalization  Resolution convert
In 'config.h'

#define AR_PATT_SAMPLE_NUM 64
#define AR_PATT_SIZE_X 16
#define AR_PATT_SIZE_Y 16

<table>
<thead>
<tr>
<th>Identification Accuracy</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large size</td>
<td>Good</td>
</tr>
<tr>
<td>Small size</td>
<td>Bad</td>
</tr>
<tr>
<td></td>
<td>Slow</td>
</tr>
<tr>
<td></td>
<td>Fast</td>
</tr>
</tbody>
</table>
2.3 Pose & Position Estimation

- Two types of initial condition
- Use of estimation accuracy
(1) Two types of initial condition

1. Geometrical calculation based on 4 vertices in screen coordinates
   double arGetTransMat( ARMarkerInfo *marker_info,
                           double center[2], double width,
                           double conv[3][4] );

2. Use of information from previous image frame
   double arGetTransMatCont( ARMarkerInfo *marker_info,
                            double prev_conv[3][4],
                            double center[2], double width,
                            double conv[3][4] );

• See 'simpleTest2.c'
(2) Use of estimation accuracy

\[
\begin{bmatrix}
    h\hat{x}_i \\
    h\hat{y}_i \\
    h
\end{bmatrix} = C \cdot T_{CM} \begin{bmatrix}
    X_{Mi} \\
    Y_{Mi} \\
    Z_{Mi} \\
    1
\end{bmatrix}, \quad i = 1,2,3,4
\]

\[
er = \frac{1}{4} \sum_{i=1,2,3,4} \left\{ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right\}
\]

arGetTransMat() minimizes the 'err'. It returns this minimized 'err'. If 'err' is still big,

- Miss-detected marker.
- Use of camera parameters by bad calibration.
2.4 Background video display

Texture mapping v.s. glDrawPixels()

- Performance depends on hardware and OpenGL driver.

external variable: argDrawMode in 'gsub.h'

#define DEFAULT_DRAW_MODE in 'config.h'

- AR_DRAW_BY_GL_DRAW_PIXELS
- AR_DRAW_BY_TEXTURE_MAPPING

Note: glDrawPixels() dose not compensate image distortion.

See 'examples/test/graphicsTest.c' and 'modeTest'
Some notes

- ARToolKit has limitation as a 3D measurement system because of monocular camera setup.
- Lighting conditions and Marker materials are also important.
Coming next

- In a half year
  - Stereo version of ARToolKit

- In a year
  - Shadow representation with VRML parser
  - Texture tracking (kind of natural feature tracking)