CV: Matching in 2D

Matching 2D images to 2D images;
Matching 2D images to 2D maps or 2D models;
Matching 2D maps to 2D maps
2D Matching

Problem

1) Need to match images to maps or models
2) need to match images to images

Applications

1) land use inventory matches images to maps
2) object recognition matches images to models
3) comparing X-rays before and after surgery
Methods for study

- Recognition by alignment
- Pose clustering
- Geometric hashing
- Local focus feature
- Relational matching
- Interpretation tree
- Discrete relaxation
Tools and methods

- Algebra of affine transformations
  scaling, rotation, translation, shear
- Least-squares fitting
- Nonlinear warping
- General algorithms
  graph-matching, pose clustering,
  discrete relaxation, interpretation tree
Alignment or registration

A mapping between 2D spaces $M$ and $I$. $M$ may be a model and $I$ an image, but in general any 2D spaces are possible.

\[
M[x, y] = I[g(x, y), h(x, y)]
\]
\[
I[r, c] = M[g^{-1}(r, c), h^{-1}(r, c)]
\]

**DEF:** Image registration is the process by which points of two images from similar viewpoints of essentially the same scene are geometrically transformed so that corresponding features of the two images have the same coordinates.
Components of transformations

- Scaling
- Rotation
- Translation
- Shear
Scaling (Affine Transformations)

\[
\begin{bmatrix}
 x' \\
 y'
\end{bmatrix} =
\begin{bmatrix}
 c & 0 \\
 0 & c
\end{bmatrix}
\begin{bmatrix}
 x \\
 y
\end{bmatrix} =
\begin{bmatrix}
 cx \\
 cy
\end{bmatrix} = c
\begin{bmatrix}
 x \\
 y
\end{bmatrix}
\]

(2)

\[
\begin{bmatrix}
 x' \\
 y'
\end{bmatrix} =
\begin{bmatrix}
 c_x & 0 \\
 0 & c_y
\end{bmatrix}
\begin{bmatrix}
 x \\
 y
\end{bmatrix} =
\begin{bmatrix}
 c_xx \\
 c_yy
\end{bmatrix}
\]

(3)

Scaling both coordinates of a 2D vector by scale factor 2.
Rotation transformation

Rotation of any 2D point in terms of rotation of the basis vectors.
Pure rotation

\[
R_\theta([x, y]) = R_\theta(x[1, 0] + y[0, 1]) \\
= xR_\theta([1, 0]) + yR_\theta([0, 1]) = x[\cos\theta, \sin\theta] + y[-\sin\theta, \cos\theta] \\
= [x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta]
\]

\[
\begin{bmatrix}
x' \\
y'
\end{bmatrix} = \begin{bmatrix}
\cos\theta & -\sin\theta \\
\sin\theta & \cos\theta
\end{bmatrix} \begin{bmatrix}
x \\
y
\end{bmatrix} = \begin{bmatrix}
x\cos\theta - y\sin\theta \\
x\sin\theta + y\cos\theta
\end{bmatrix} \tag{4}
\]
Orthogonal transformations

- DEF: set of vectors is orthogonal if all pairs are perpendicular
- DEF: set of vectors is orthonormal if it is orthogonal and all vectors have unit length
- Orthogonal transformations preserve angles
- Orthonormal transformations preserve angles and distances
- Rotations and translations are orthonormal
- DEF: a rigid transformation is combined rotation and translation
Translation requires homogeneous coordinates

**Definition** The homogeneous coordinates of a 2D point \( P = [x, y]^t \) are \([sx, sy, s]^t\), where \( s \) is a scale factor, commonly 1.0.

**Matrix Representation of a 2D Translation**

\[
\begin{bmatrix}
x' \\
y' \\
1
\end{bmatrix} = \begin{bmatrix}
1 & 0 & x_0 \\
0 & 1 & y_0 \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
x \\
y \\
1
\end{bmatrix} = \begin{bmatrix}
x + x_0 \\
y + y_0 \\
1
\end{bmatrix}
\]
Model of shear

Map the basis vectors \([1, 0], [0, 1]\) onto \([1, e_u], [0, 1]\) and/or Map the basis vectors \([1, 0], [0, 1]\) onto \([1, 0], [e_u, 1]\)

Figure 1: (Left) \(v\)-axis shear and (right) \(u\)-axis shear.

\[
\begin{bmatrix}
 x \\
 y \\
 1
\end{bmatrix} =
\begin{bmatrix}
 1 & 0 & 0 \\
 e_u & 1 & 0 \\
 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
 u \\
 v \\
 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
 x \\
 y \\
 1
\end{bmatrix} =
\begin{bmatrix}
 1 & 0 & 0 \\
 e_v & 1 & 0 \\
 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
 u \\
 v \\
 1
\end{bmatrix}
\]

http://blog.sciencenet.cn/blog-47505-446537.html
General affine transformation

1. We already covered scaling, rotation, translation, shearing; now we add reflection.

2. Reflection about the u-axis maps \([1,0], [0,1]\) onto \([1,0], [0,-1]\) respectively.

3. Reflection about the v-axis maps \([1,0], [0,1]\) onto \([-1,0], [0,1]\) respectively.

4. All operations scaling, rotation, translation, shearing, reflection (in 2D) have inverses of the same type (e.g. can be undone).

5. ANY COMBINATION of scaling, rotation, translation, shearing, reflection is modeled by matrix multiplication—with the combined form:

\[
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix} =
\begin{bmatrix}
  a_{11} & a_{12} & a_{13} \\
  a_{21} & a_{22} & a_{23} \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  u \\
  v \\
  1
\end{bmatrix}
\]

6. Ideally, these 6 parameters can be determined by matching (registering) 3 points in 2 images. In practice, a least squares solution from matching many more points gives a more robust registration formula—SEE LATER SLIDE.
Solving for an RST using control points

Assume two corresponding control points \( P_1 \) and \( P_2 \)

1. determine rotation \( \theta \)
   
   (a) direction of the vector \( P_1P_2 \) in \( I \) is determined as:
   
   \[
   \theta_i = \arctan((y_2 - y_1)/(x_2 - x_1))
   \]
   
   (b) direction of the vector in \( W \) is determined as:
   
   \[
   \theta_w = \arctan((y_2 - y_1)/(x_2 - x_1))
   \]
   
   (c) \( \theta = \theta_w - \theta_i \).

2. \( \theta \) is determined: all \( \sin \) and \( \cos \) elements are known.

3. solve 3 equations and 3 unknowns for \( s \) and \( x_0, y_0 \).

\[
\begin{bmatrix}
  x_w \\
  y_w \\
  1
\end{bmatrix}
= \begin{bmatrix}
  1 & 0 & x_0 \\
  0 & 1 & y_0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  s & 0 & 0 \\
  0 & s & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  \cos \theta & -\sin \theta & 0 \\
  \sin \theta & \cos \theta & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x_i \\
  y_i \\
  1
\end{bmatrix}
\]

\[
x_w = x_i s \cos \theta - y_i s \sin \theta + x_0
\]

\[
y_w = x_i s \sin \theta + y_i s \cos \theta + y_0
\]
Extracting a subimage by subsampling

Distorted face of Andrew Jackson extracted from a $20$ bill by defining an affine mapping with shear.
Subsampling transformation

\[
\begin{align*}
\begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \frac{r}{n} \left( \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} - \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \right) + \frac{c}{m} \left( \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} - \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \right) \\
\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} &= \begin{bmatrix} (x_1 - x_0)/n & (x_2 - x_0)/m & x_0 \\ (y_1 - y_0)/n & (y_2 - y_0)/m & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r \\ c \\ 1 \end{bmatrix}
\end{align*}
\]

\( [r, c] \) subscripts the output image; \([x, y]\) subscripts the input image. The output image has \( n \) rows and \( m \) columns. The mapping computes, for each \([r, c]\) of output where to access the intensity in the input \([x, y]\).
At MSU, even the pigs are smart.
recognition by alignment

- Automatically match some salient points
- Derive a transformation based on the matching points
- Verify or refute the match using other feature points
- If verified, then registration is done, else try another set of matching points
Recognition by alignment
Feature points and distances

<table>
<thead>
<tr>
<th>point</th>
<th>coordinates</th>
<th>to A</th>
<th>to B</th>
<th>to C</th>
<th>to D</th>
<th>to E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(8,17)</td>
<td>0</td>
<td>12</td>
<td>15</td>
<td>37</td>
<td>21</td>
</tr>
<tr>
<td>B</td>
<td>(16,26)</td>
<td>12</td>
<td>0</td>
<td>12</td>
<td>30</td>
<td>26</td>
</tr>
<tr>
<td>C</td>
<td>(23,16)</td>
<td>15</td>
<td>12</td>
<td>0</td>
<td>22</td>
<td>15</td>
</tr>
<tr>
<td>D</td>
<td>(45,20)</td>
<td>37</td>
<td>30</td>
<td>22</td>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>E</td>
<td>(22,1)</td>
<td>21</td>
<td>26</td>
<td>15</td>
<td>30</td>
<td>0</td>
</tr>
</tbody>
</table>
Image features pts and distances

Detected Features

<table>
<thead>
<tr>
<th>point</th>
<th>coordinates</th>
<th>to $H_1$</th>
<th>to $H_2$</th>
<th>to $H_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_1$</td>
<td>(31,9)</td>
<td>0</td>
<td>21</td>
<td>26</td>
</tr>
<tr>
<td>$H_2$</td>
<td>(10,12)</td>
<td>21</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>$H_3$</td>
<td>(10,24)</td>
<td>26</td>
<td>12</td>
<td>0</td>
</tr>
</tbody>
</table>
Point matches reflect distances
Once matching bases fixed

- can find any other feature point in terms of the matching transformation
- can go back into image to explore for the holes that were missed (C and D)
- can determine grip points for a pick and place robot (transform R and Q into the image coordinates)
Once we have matching control points \((H_2, A)\) and \((H_3, B)\) we can compute a rigid transform.
Get rotation easily, then translation

1. Compute $\theta$ from the difference in heading
   The direction of the vector from $A$ to $B$ in the model is $\theta_1 = \arctan(9.0/3.0) = 0.844$ and the heading of the corresponding vector from $H_2$ to $H_3$ in the image is $\theta_2 = \arctan(12.0/0.0) = \pi/2 = 1.571$. The rotation is thus $\theta = 0.727$ radians.

2. Distance matrices show that scale $s = 1$

3. Compute translation $[u_0, v_0]$

   \[
   \begin{bmatrix}
   u \\
   v \\
   1
   \end{bmatrix} =
   \begin{bmatrix}
   10 \\
   12 \\
   1
   \end{bmatrix} =
   \begin{bmatrix}
   \cos \theta & -\sin \theta & u_0 \\
   \sin \theta & \cos \theta & v_0 \\
   0 & 0 & 1
   \end{bmatrix}
   \begin{bmatrix}
   8 \\
   17 \\
   1
   \end{bmatrix}
   \]

   The two resulting linear equations readily produce $u_0 = 15.3$ and $v_0 = -5.95$.  

MSU CSE 803
Assume affine transformation from model into image space. Compute transforms with significant supporting evidence.

1. process image to obtain items of evidence $E_j$
2. for each item of evidence $E_j$, vote for registration parameters $c_k$ that it supports
3. significant clusters in $\alpha$-space yield registration of model to image data
“Best” affine transformation from overdetermined matches

A general affine transformation from 2D to 2D requires six parameters and can be computed from only 3 matching pairs of points \((x_j, y_j), (u_j, v_j)\) \(j=1,2\).

\[
\begin{bmatrix}
    u \\
    v \\
    1
\end{bmatrix} =
\begin{bmatrix}
    a_{11} & a_{12} & a_{13} \\
    a_{21} & a_{22} & a_{23} \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    x \\
    y \\
    1
\end{bmatrix}
\] (14)

A Least-Squares approach will be more robust.

\[
\varepsilon(a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}) = \sum_{j=1}^{n} \left( (a_{11}x_j + a_{12}y_j + a_{13} - u_j)^2 + (a_{21}x_j + a_{22}y_j + a_{23} - v_j)^2 \right) (15)
\]
“Best” affine transformaiton

A Least-Squares approach will be more robust.

\[ \varepsilon(a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}) = \sum_{j=1}^{n} \left( (a_{11}x_j + a_{12}y_j + a_{13} - u_j)^2 + (a_{21}x_j + a_{22}y_j + a_{23} - v_j)^2 \right) \] (15)

Taking the six partial derivatives of the error function with respect to each of the six variables and setting this expression to zero gives us the six equations represented in matrix form.

\[
\begin{bmatrix}
\Sigma x_j^2 & \Sigma x_j y_j & \Sigma x_j & 0 & 0 & 0 \\
\Sigma x_j y_j & \Sigma y_j^2 & \Sigma y_j & 0 & 0 & 0 \\
\Sigma x_j & \Sigma y_j & \Sigma 1 & 0 & 0 & 0 \\
0 & 0 & 0 & \Sigma x_j^2 & \Sigma x_j y_j & \Sigma x_j \\
0 & 0 & 0 & \Sigma x_j y_j & \Sigma y_j^2 & \Sigma y_j \\
0 & 0 & 0 & \Sigma x_j & \Sigma y_j & \Sigma 1 \\
\end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \end{bmatrix} = \begin{bmatrix} \Sigma u_j x_j \\ \Sigma u_j y_j \\ \Sigma u_j \\ \Sigma v_j x_j \\ \Sigma v_j y_j \\ \Sigma v_j \end{bmatrix} \] (16)

Use as many matching pairs \(((x,y)(u,v))\) as possible
result for previous town match

Matching control point pairs are:

<table>
<thead>
<tr>
<th>288</th>
<th>210</th>
<th>31</th>
<th>160</th>
<th>232</th>
<th>288</th>
<th>95</th>
<th>205</th>
<th>195</th>
<th>372</th>
<th>161</th>
<th>229</th>
<th>269</th>
<th>314</th>
<th>112</th>
<th>159</th>
</tr>
</thead>
<tbody>
<tr>
<td>203</td>
<td>424</td>
<td>199</td>
<td>209</td>
<td>230</td>
<td>336</td>
<td>130</td>
<td>196</td>
<td>284</td>
<td>401</td>
<td>180</td>
<td>124</td>
<td>327</td>
<td>428</td>
<td>198</td>
<td>69</td>
</tr>
<tr>
<td>284</td>
<td>299</td>
<td>100</td>
<td>146</td>
<td>337</td>
<td>231</td>
<td>45</td>
<td>101</td>
<td>369</td>
<td>223</td>
<td>38</td>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Transformation Matrix is:

\[
\begin{bmatrix}
-0.0414 & 0.773 & -119 \\
-1.120 & -0.213 & 526 \\
0.0 & 0.0 & 1.0
\end{bmatrix}
\]

Residuals (in pixels) for 22 equations are as follows:

<table>
<thead>
<tr>
<th>0.18</th>
<th>-0.68</th>
<th>-1.22</th>
<th>0.47</th>
<th>-0.77</th>
<th>0.06</th>
<th>0.34</th>
<th>-0.51</th>
<th>1.09</th>
<th>0.04</th>
<th>0.96</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.51</td>
<td>-1.04</td>
<td>-0.81</td>
<td>0.05</td>
<td>0.27</td>
<td>0.13</td>
<td>-1.12</td>
<td>0.39</td>
<td>-1.04</td>
<td>-0.12</td>
<td>1.81</td>
</tr>
</tbody>
</table>
Registering Image $^{1}I_{t1}$ to $^{2}I_{t2}$
Registering Image $^{1}I_{t1}$ to $^{2}I_{t2}$

11 matching control points $x, y, u, v$ below

--------Best 2D Affine Fit Program--------

Matching control point pairs are:

```
<p>| | | | | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>288</td>
<td>210</td>
<td>31</td>
<td>160</td>
<td>232</td>
<td>288</td>
<td>95</td>
<td>205</td>
<td>195</td>
<td>372</td>
<td>161</td>
<td>229</td>
<td>269</td>
<td>314</td>
<td>112</td>
<td>159</td>
</tr>
<tr>
<td>203</td>
<td>424</td>
<td>199</td>
<td>209</td>
<td>230</td>
<td>336</td>
<td>130</td>
<td>196</td>
<td>284</td>
<td>401</td>
<td>180</td>
<td>124</td>
<td>327</td>
<td>428</td>
<td>198</td>
<td>69</td>
</tr>
<tr>
<td>284</td>
<td>299</td>
<td>100</td>
<td>146</td>
<td>337</td>
<td>231</td>
<td>45</td>
<td>101</td>
<td>369</td>
<td>223</td>
<td>38</td>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

The Transformation Matrix is:

\[
\begin{bmatrix}
-0.0414 & 0.773 & -119 \\
-1.120 & -0.213 & 526 \\
0.0 & 0.0 & 1.0
\end{bmatrix} = T
\begin{bmatrix}
u \\
v \\
1
\end{bmatrix} = T
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}^t
\]
Least Squares in MATLAB

BigPic =

| 288 | 210 | 1 |
| 203 | 424 | 1 |
| 284 | 299 | 1 |
| 232 | 288 | 1 |
| 230 | 336 | 1 |
| 337 | 231 | 1 |
| 195 | 372 | 1 |
| 284 | 401 | 1 |
| 369 | 223 | 1 |
| 269 | 314 | 1 |
| 327 | 428 | 1 |

LilPic =

| 31  | 160 |
| 199 | 209 |
| 100 | 146 |
| 95  | 205 |
| 130 | 196 |
| 45  | 101 |
| 161 | 229 |
| 180 | 124 |
| 38  | 64  |
| 112 | 159 |
| 198 | 69  |

\[
\begin{bmatrix}
a_{11} & a_{21} \\
a_{12} & a_{22} \\
a_{13} & a_{23}
\end{bmatrix}
\]
Least squares in MATLAB 2

```matlab
>> AFFINE = BigPic \ LilPic

AFFINE =
-0.0414  -1.1203
0.7728   -0.2126
-119.1931 526.6200

>> ERROR = LilPic - BigPic * AFFINE

ERROR =
-0.1855  0.6821
-1.0863  -0.0446
-0.1323  1.1239
 1.2175  -0.4715
-0.9608  -1.5064
-0.3877  1.0428
 0.7696  -0.0633
 1.0396   0.8111
 0.1188  -1.8080
-0.3452   0.5084
-0.0477  -0.2745
```

Worst is 1.8 pixels

The solution is such that the 11D vector at the right has the smallest L2 norm.
# Least squares in MATLAB 3

- X =
  
  288 210 1
  203 424 1
  284 299 1
  232 288 1

- >> Y
  
  Y =

  31 160
  199 209
  100 146
  95 205

- >> T = X \ Y
  
  T =

  -0.0620  -1.0917
  0.7592   -0.2045
  -109.8863 517.2541

- >> E = Y - X*T
  
  E =

  -0.6846  0.0974
  -0.4327  0.0616
  0.4959  -0.0706
  0.6214  -0.0884

Solution from 4 points has smaller error on those points.
When the affine transformation obtained from 4 matching points is applied to all 11 points, the error is much worse than when the transformation was obtained from those 11 points.

\[
\text{>> } E2 = \text{LilPic} - \text{BigPic} \times T
\]

\[
E2 =
\begin{bmatrix}
-0.6846 & 0.0974 \\
-0.4327 & 0.0616 \\
0.4959 & -0.0706 \\
0.6214 & -0.0884 \\
-0.9456 & -1.4568 \\
0.4115 & -1.1150 \\
0.5508 & 0.6949 \\
3.0546 & -1.2135 \\
1.4706 & -4.8164 \\
0.1769 & -0.3789 \\
3.2231 & -3.7493
\end{bmatrix}
\]