2D matching part 2

• Review of alignment methods and errors in using them

• Introduction to more 2D matching methods
Review of roadmap: algorithms to control matching

1. recognition by alignment (Part I)
2. pose clustering
3. geometric hashing
4. local focus feature matching
5. relational matching
6. interpretation tree
7. discrete relaxation
Rigid transformation review

- Rigid Transformation
  1. scale factor is uniform and 1.0
  2. any rotation
  3. any translation
  4. determine mapping from 2 matching pairs \((P_1, Q_1), (P_2, Q_2)\)
  5. **Problem:** error at other points can be bad
Affine includes scaling and shear

- General Affine Transformation
  1. scaling $s_x$ and $s_y$
  2. any rotation $\theta$
  3. any translation $(x_0, y_0)$
  4. any shear in $x$ or $y$
  5. determine mapping from 3 matching pairs $(P_1, Q_1), (P_2, Q_2), (P_3, Q_3)$
  6. Problem: error at other points can be bad
Problems with error

- Least squares fitting uses $n >> 3$ point pairs
- Significantly reduces error across field
- Will still be thrown off by outliers
  * can throw out pairs with high error and then refit
  * can set the “weight” of any pair to be inversely proportional to error squared
Sources of error

- image quantization (Hough parameter quantization, etc.)
- feature detection error in location or orientation
- stretching of film, if used
- lens distortion
- roll or pitch of camera (airplane, robot, etc.)

* Wrong matching in the pair of points yields outlier
Remove outlier and refit

Model Points $P_j$
Image Points $Q_k$
Sometimes a hallucination

6 points match, but the objects do not. Can verify using more model points.
Pose clustering (generalized Hough transform)

- Use $m$ minimal sets of matching features, each just enough to compute alignment
- Vector of alignment parameters is put as evidence into “parameter space”
- When all $m$ units of evidence computed, examine parameter space for cluster[s]
Compute matching transformation using pose-clustering.
Let $P_i, i = 1, D$ be the detected image features.
Let $L_j, j = 1, M$ be the stored model features.

```
procedure pose_clustering(P,L);
{
for each pair of image feature points $(P_i, P_j)$
    for each pair of model feature points $(L_m, L_n)$ of same type
    {
      compute parameters $\alpha$ of RST mapping
      pair $(L_m, L_n)$ onto $(P_i, P_j)$;
      contribute $\alpha$ to the cluster space;
    }
    examine space of all candidates $\alpha$ for clusters;
verify every large cluster by mapping all
    model feature points and checking the image;
return(verified $\{x_k\}$);
}
```
Line segment junctions for matching

'L' junction  'Y' junction  'T' junction  'Arrow' junction  'X' junction
“Abstract vectors” subtending detected junctions

Abstract vector with tail at T and tip at Y, or tail at L and tip at X
Parameter space resulting from 10 vector matches

<table>
<thead>
<tr>
<th>Model Pair</th>
<th>Image Pair</th>
<th>θ</th>
<th>s</th>
<th>u₀</th>
<th>v₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>L(170,220),X(100,200)</td>
<td>L(545,400),X(200,120)</td>
<td>0.403</td>
<td>6.10</td>
<td>118</td>
<td>-1240</td>
</tr>
<tr>
<td>L(170,220),X(100,200)</td>
<td>L(420,370),X(360,500)</td>
<td>5.14</td>
<td>2.05</td>
<td>-97</td>
<td>514</td>
</tr>
<tr>
<td>T(100,100),Y(40,150)</td>
<td>T(260,240),Y(100,245)</td>
<td>0.663</td>
<td>2.05</td>
<td>225</td>
<td>-48</td>
</tr>
<tr>
<td>T(100,100),Y(40,150)</td>
<td>T(140,380),Y(300,380)</td>
<td>3.87</td>
<td>2.05</td>
<td>166</td>
<td>669</td>
</tr>
<tr>
<td>L(200,100),X(220,170)</td>
<td>L(545,400),X(200,120)</td>
<td>2.53</td>
<td>6.10</td>
<td>1895</td>
<td>200</td>
</tr>
<tr>
<td>L(200,100),X(220,170)</td>
<td>L(420,370),X(360,500)</td>
<td>0.711</td>
<td>1.97</td>
<td>250</td>
<td>-36</td>
</tr>
<tr>
<td>L(260,70),X(40,70)</td>
<td>L(545,400),X(200,120)</td>
<td>0.682</td>
<td>2.02</td>
<td>226</td>
<td>-41</td>
</tr>
<tr>
<td>L(260,70),X(40,70)</td>
<td>L(420,370),X(360,500)</td>
<td>5.14</td>
<td>0.651</td>
<td>308</td>
<td>505</td>
</tr>
<tr>
<td>T(150,125),Y(150,50)</td>
<td>T(260,240),Y(100,245)</td>
<td>4.68</td>
<td>2.13</td>
<td>3</td>
<td>568</td>
</tr>
<tr>
<td>T(150,125),Y(150,50)</td>
<td>T(140,380),Y(300,380)</td>
<td>1.57</td>
<td>2.13</td>
<td>407</td>
<td>60</td>
</tr>
</tbody>
</table>

Rotation, scale, translation computed as in single match alignment. Use the cluster center to estimate best alignment parameters.
Detecting airplanes on airfield

(a) original airfield image
Airplane model of abstract vectors; detected image features

(b) model of object

(c) detections matching model
Relational matching method

- A **part** is an object or structure in the scene such as region segment, edge segment, hole, corner or blob.

- A **label** is a symbol assigned to a part to identify/recognize it at some level.

- An **assignment** is a mapping from parts to labels.
Some relations between parts

- adjacent\((P_1, P_2)\)
- inside\((P_1, P_2)\)
- connectsto\((P_1, P_2)\)
- toposf\((P_1, P_2)\)
- leftof\((P_1, P_2)\)
1 Definition Given a set of parts \( P \), a set of labels for those parts \( L \), a relation \( R_P \) over \( P \), and a second relation \( R_L \) over \( L \), a consistent labeling \( f \) is an assignment of labels to parts that satisfies:

\[
\text{If } (p_i, p_i') \in R_P, \text{ then } (f(p_i), f(p_i')) \in R_L.
\]
Parts, labels, relations

\[ P = \{S_1, S_2, S_3, S_4, S_5, S_6, S_7, S_8, S_9, S_{10}, S_{11}\}. \]

\[ L = \{S_a, S_b, S_c, S_d, S_e, S_f, S_g, S_h, S_i, S_j, S_k, S_l, S_m\}. \]

\[ R_P = \{ (S_1, S_2), (S_1, S_5), (S_1, S_6), (S_2, S_3), (S_2, S_4), (S_3, S_4), (S_3, S_9), (S_4, S_5), (S_4, S_7), (S_4, S_{11}), (S_5, S_6), (S_5, S_7), (S_5, S_{11}), (S_6, S_8), (S_6, S_{11}), (S_7, S_9), (S_7, S_{10}), (S_7, S_{11}), (S_8, S_{10}), (S_8, S_{11}), (S_9, S_{10}) \}. \]

\[ R_L = \{ (S_a, S_b), (S_a, S_j), (S_a, S_n), (S_b, S_c), (S_b, S_d), (S_b, S_n), (S_c, S_d), (S_d, S_e), (S_d, S_f), (S_d, S_g), (S_e, S_f), (S_e, S_g), (S_f, S_l), (S_f, S_m), (S_g, S_h), (S_g, S_i), (S_g, S_n), (S_h, S_i), (S_h, S_k), (S_h, S_l), (S_h, S_n), (S_i, S_j), (S_i, S_k), (S_i, S_n), (S_j, S_k), (S_k, S_l), (S_l, S_m) \}. \]
In a consistent labeling “image” parts relate as do “model” parts

<table>
<thead>
<tr>
<th>$f(S1)$</th>
<th>$= \quad S_j$</th>
<th>$f(S7)$</th>
<th>$= \quad S_g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(S2)$</td>
<td>$= \quad S_a$</td>
<td>$f(S8)$</td>
<td>$= \quad S_l$</td>
</tr>
<tr>
<td>$f(S3)$</td>
<td>$= \quad S_b$</td>
<td>$f(S9)$</td>
<td>$= \quad S_d$</td>
</tr>
<tr>
<td>$f(S4)$</td>
<td>$= \quad S_n$</td>
<td>$f(S10)$</td>
<td>$= \quad S_f$</td>
</tr>
<tr>
<td>$f(S5)$</td>
<td>$= \quad S_i$</td>
<td>$f(S11)$</td>
<td>$= \quad S_h$</td>
</tr>
<tr>
<td>$f(S6)$</td>
<td>$= \quad S_k$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Distance relation often used

12(P₁, P₂) means that \( d(P₁, P₂) = 12 \)

(Left) Model object and (right) three holes detected in an image.
Model Point Locations and Interpoint Distances
(Left) Model object and (right) three holes detected in an image.

Model Point Locations and Interpoint 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 Point Locations and Interpoint Distances

<table>
<thead>
<tr>
<th>point</th>
<th>coordinates †</th>
<th>to H1</th>
<th>to H2</th>
<th>to H3</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1</td>
<td>(31,9)</td>
<td>0</td>
<td>21</td>
<td>26</td>
</tr>
<tr>
<td>H2</td>
<td>(10,12)</td>
<td>21</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>H3</td>
<td>(10,24)</td>
<td>26</td>
<td>12</td>
<td>0</td>
</tr>
</tbody>
</table>
Partial *Interpretation Tree* to find a distance consistent labeling

The IT shows matching attempts that can be tried using a backtracking algorithm. If a relation fails the algorithm tries a different branch.

```
Partial Interpretation Tree to find a distance consistent labeling

The IT shows matching attempts that can be tried using a backtracking algorithm. If a relation fails the algorithm tries a different branch.
```
Detailed IT algorithm

Let $P$ be the set of detected image features.
Let $L$ be the set of stored model features.
Let $R_P$ be the relationship over the image features.
Let $R_L$ be the relationship over the model features.
Let $f$ be the consistent labeling to be returned, initially NIL.

Current matching pairs can be stored in the recursive stack. If a new pair is consistent with the previous pairs, continue forward; if not, then back up (and retract the recent pairing).
procedure Interpretation_Tree_Search($P, L, R_P, R_L, f$);
{
    $p := \text{first}(P)$;
    for each $l$ in $L$
    {
        $f' = f \cup \{(p, l)\}$; /* add part-label to interpretation */
        OK = true;
        for each N-tuple $(p_1, \ldots, p_N)$ in $R_P$ containing component $p$
            and whose other components are all in $\text{domain}(f)$
            /* check on relations */
            if $(f(p_1), \ldots, f(p_N))$ is not in $R_L$ then
            {
                OK: = false;
                break;
            }
        if OK then
        {
            $P' = \text{rest}(P)$;
            if isempty($P'$) then output($f'$);
            else Interpretation_Tree_Search($P', L, R_P, R_L, f'$);
        }
    }
}
Discrete relaxation labeling constrains possible labels

- **initialize**: each image feature labeled by set of all compatible model labels
- **iterate**: delete labels that are inconsistent with known relations
- **stop**: when no label sets change

A sometimes useful method that once drew much interest (see pubs by Rosenfeld, Zucker, Hummel, etc.) The Marr-Poggio stereo matching algorithm has the character of relaxation.
Discrete relaxation labeling constrains possible labels

Let $P_i, i = 1, D$ be the detected image features.
Let $S(P_i), i = 1, D$ be the set of initially compatible labels.

```plaintext
procedure Relaxation_Labeling(P, S);
{
repeat
  for each $(P_i, S(P_i))$ do in parallel
  {
    for each label $L_k \in S(P_i)$
      for each relation $R(P_i, P_j)$ in the image parts
        if $\exists L_m \in S(P_j)$ with $R(L_k, L_m)$ in model
          then keep $L_k$ in $S(P_i)$
        else delete $L_k$ from $S(P_i)$
  }
  until no change in any set $S(P_i)$
return(S);
}
```
Kleep matching via relaxation

Label Sets Midway through first pass of relaxation labeling.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_1$</td>
<td>$\text{no } N \ni d(A, N) = 26$</td>
<td>$\text{no } N \ni d(B, N) = 21$</td>
<td>$\text{no } N \ni d(C, N) = 26$</td>
<td>possible</td>
<td>possible</td>
</tr>
<tr>
<td>$H_2$</td>
<td>$21(H_2, H_1)$ $E \in L(H_1)$ $12(H_2, H_3)$ $B \in L(H_3)$</td>
<td>$\text{no } N \ni d(B, N) = 21$</td>
<td>$21(H_2, H_1)$ $D \in L(H_1)$ $12(H_2, H_3)$ $B \in L(H_3)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$H_1$</td>
<td>$\text{no } N \ni d(A, N) = 26$</td>
<td>$12(H_3, H_2)$ $A \in L(H_2)$ $26(H_3, H_1)$ $E \in L(H_1)$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Removing a possible label for one part affects labels for related parts

After completion of the second pass of relaxation labeling

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_1$</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>possible</td>
</tr>
<tr>
<td>$H_2$</td>
<td>possible</td>
<td>no</td>
<td>possible</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>$H_3$</td>
<td>no</td>
<td>possible</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

After completion of the third pass of relaxation labeling

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_1$</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>possible</td>
</tr>
<tr>
<td>$H_2$</td>
<td>possible</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>$H_3$</td>
<td>no</td>
<td>possible</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>
Relaxation labeling

- Can work truly in parallel
- Pairwise constraints are weaker than what the IT method can check, so sometimes the IT must follow the relaxation method
- There is “probabalistic relaxation” which changes probability of labels rather than just keeping or deleting them
- Relaxation was once thought to model human visual processes.