Announcing AVA: A Finely Labeled Video Dataset for Human Action Understanding
Thursday, October 19, 2017
Posted by Chunhui Gu & David Ross, Software Engineers
Today

Mid-term review
Intro to Stereo Vision
Dense Stereo
Depth from Disparity
Camera Calibration
We have an Source group of 1D points $s_i$ and a Target group $t_i$, and we're looking to find the best transform parameters $x_1, x_2$ (scale, translation).

This is the minimization scheme:

$$x_1, x_2 = \arg\min_{x_1, x_2} E(S, T, X) = \sum_{i=1}^{n} (t_i - s_i x_1 - x_2)^2$$

To solve we need to take the derivative w.r.t $x$:

Your goal is to solve for $X$ (calculate the derivative and isolate $X$).

Write your answer in the form: $X = \ldots$, where on the r.h.s you have only $S, T$, i.e. $X = f(S, T)$. 

We can flesh out the functional using vector/matrix multiplication notation:

$$E(S, T, X) = \sum_{i=1}^{n} \left| \begin{pmatrix} s_i & 1 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - t_i \right|^2 = \|SX - T\|^2 = \ldots$$

$$= T^T T - 2(SX)^T T + (SX)^T (SX)$$
Minimization term:

\[ E(S,T,X) = \sum_{i=1}^{n} (s_i \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - t_i)^2 = \|SX - T\|^2 = \ldots \]

\[ = T^T T - 2(SX)^T T + (SX)^T (SX) \]

Take derivative w.r.t \( X \):

\[ \frac{\partial E(S,T,X)}{\partial X} = 0 \]

Remember

\[ \frac{\partial aX}{\partial X} = a \quad , \quad \frac{\partial a}{\partial X} = 0 \quad , \quad \frac{\partial X^T X}{\partial X} = 2X \quad , \quad \frac{\partial X^T AX}{\partial X} = 2AX \]

\[ \frac{\partial E(X,S,T)}{\partial X} = \frac{\partial [T^T T - 2(SX)^T T + (SX)^T (SX)]}{\partial X} = 0 \]

\[ \frac{\partial [T^T T - 2X^T S^T T + X^T S^T S X]}{\partial X} = \ldots \]

\[ 0 - 2S^T T + 2S^T SX = 0 \]

\[ 2S^T SX = 2S^T T \]

\[ X = (S^T S)^{-1} S^T T \]
### What are the two fundamental differences between generative and discriminative models?

- **Generative models have a model for the class probability given the data** $P(c|x)$, **while Discriminative models have a model for the data given the class** $P(x|c)$.

| Generative models have a model for the class probability given the data $P(c|x)$, while Discriminative models have a model for the data given the class $P(x|c)$ |
|---|
| Discriminative models find boundaries in the data between classes (e.g. SVM), while Generative models use a probabilistic model for the data and usually make a decision with a threshold (e.g. Naive Bayes) |

| Generative models have a model for the class probability given the data $P(c|x)$, while Discriminative models have a model for the data given the class $P(x|c)$ |
|---|
| Discriminative models find boundaries in the data between classes (e.g. SVM), while Generative models use a probabilistic model for the data and usually make a decision with a threshold (e.g. Naive Bayes) |

| Generative models have a model for the class probability given the data $P(c|x)$, while Discriminative models have a model for the data given the class $P(x|c)$ |
|---|
| Discriminative models find boundaries in the data between classes (e.g. SVM), while Generative models use a probabilistic model for the data and usually make a decision with a threshold (e.g. Naive Bayes) |

| Generative models have a model for the class probability given the data $P(c|x)$, while Discriminative models have a model for the data given the class $P(x|c)$ |
|---|
| Discriminative models find boundaries in the data between classes (e.g. SVM), while Generative models use a probabilistic model for the data and usually make a decision with a threshold (e.g. Naive Bayes) |

<table>
<thead>
<tr>
<th>Generative models can generate data by only using class probabilities (e.g. a histogram), while Discriminative models cannot generate data since they don't use supervised learning (annotated datasets)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discriminative models find boundaries in the data between classes (e.g. SVM), while Generative models use a probabilistic model for the data and usually make a decision with a threshold (e.g. Naive Bayes)</td>
</tr>
</tbody>
</table>
What are Multi-scaled Oriented Patches? (MOPS)

- Feature descriptors that are simple image patches that were normalized w.r.t: Orientation, Scale, and Intensity.

- Feature descriptors using image patches that have the shape of a mop.

- Feature descriptors using image patches that are only scale invariant.

- Feature descriptors using image patches that were normalized w.r.t Scale and Orientation.
What does the Particle Filter (PF) offer in terms of tracking?

- Keep multiple simultaneous hypotheses of the object's location. An implicit model of the object's position PDF, which can model very complex PDFs.

- Keep multiple simultaneous hypotheses of the object's location. An explicit model of the object's position PDF, e.g. Normal, Uniform or multi-modal.

- Keep multiple simultaneous hypotheses of the object's location. An implicit model of the object's position PDF, which can model very complex PDFs. A guarantee of convergence around the object position.

- Keep just a single hypotheses of the object's location, but a very accurate one. An implicit model of the object's position PDF, which can model very complex PDFs.
Histograms of pixel intensities are invariant to ...

- Rotation
  Translation
  Scale (if they are normalized)

- Rotation
  Scale (if they are normalized)

- Rotation
  Translation

- Histograms of intensities are not particularly invariant.

- Translation
  Scale (if they are normalized)
Stereo Vision
Epipolar Constraint

Epipolar constraint:

\[ x_L^T Ex_r = 0 \]

How did we get that?

Take a vector perpendicular to the epipolar plane: \( t \times (Rx) \)

A vector to the other point should be perpendicular to it (dot product = 0):

\[ x \cdot [t \times (Rx)] = 0 \]
Epipolar Constraint

Calibrated camera: $K = I$

Calibrated camera: $K = I$

$x_L = [I \ 0](x,1)^T = x$

$x_R = [R \ t](x,1)^T = Rx + t$
Epipolar Constraint

Perpendicular to epipolar plane

$O_L (0,0,0) \quad e_L \quad X_L \quad X \quad (x,1)^T \quad X_1 \quad X_2 \quad X_3 \quad e_R \quad O_R \quad Rx \quad t \times Rx \quad Rx + t$
Epipolar Constraint

\[(Rx + t) \cdot [t \times Rx] = 0 \Rightarrow x_R^T([t]_x R)x = 0 \Rightarrow x_R^TEx_L = 0\]
Parallel Motion

Simple case:

Rotation is identity, motion parallel to image plane.

Y-coordinate is the same in both images

\[
R = I_{3 \times 3} \quad \text{and} \quad t = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T
\]

\[
E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}
\]

Epipolar lines are axis-aligned.
Disparity
Disparity

Easier to see like this:

\[ d_2 > d_1 \]

Insight?
In parallel motion horizontal disparity is proportional to depth.

Disparity: \( d = EF + GH \)

\[ \Rightarrow EF + GH = \left( b \cdot f \right) / z \]

**Insight:** we should strive for horizontal disparity (parallel motion).

Remember the epipolar constraint: **Look for match along the epipolar line.**

In the “flat” epilines case: **Look over x axis.**
Rectification

Make the epipolar lines parallel in both images

Calculate Homographies $\Phi_1$, $\Phi_2$:

- Start with $\Phi_2$ which breaks down as $\Phi_2 = T_3 T_2 T_1$
  - Move origin to center of image

$$T_1 = \begin{bmatrix} 1 & 0 & -\delta_x \\ 0 & 1 & -\delta_y \\ 0 & 0 & 1 \end{bmatrix}$$

- Rotate epipole to horizontal direction

$$T_2 = \begin{bmatrix} \cos[-\theta] & -\sin[-\theta] & 0 \\ \sin[-\theta] & \cos[-\theta] & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Move epipole to infinity

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1/c_x & 0 & 1 \end{bmatrix}$$
Rectification

Before
Rectification

After
Preprocess: Rectify both images

For each pixel do:

1. Scan epilines (x axis)
2. Find best match (e.g. min ||I(p_1) - J(p_2)||)
3. Compute disparity \(d = x_2 - x_1\)
4. Calculate depth \(z = (b \cdot f)/d\)
5. Repeat

Assume Small baseline:

- Appearance change is minimal
- Less occlusion
- Rectifying homographies do not degenerate (like that panorama we saw)
Improvement: Slide a **window**

Take best point: SSD, norm-corr over window.

**When does it fail?**

Hint: think of assumptions
Failures:

- Not enough) texture
  - Low contrast?
- Occlusions
- Repetitions (confusing window slide)
- Motion:
  - Very large motion
  - Non-camera motion: moving objects
- Lighting problems: Specular, shade, etc
  - Violations of Brightness Constancy
- Camera calibration errors

Parametric:

- Window size
  - Smaller - more detail, more noise
  - Bigger - less detail, less noise
- Slide region
Window Methods
Questions?
Global Constraints

So far: Each window matched independently

Assumptions:

- Brightness Constancy:
  Matching pixels will have a high match score.
  e.g. low SSD, $||I(p_i) - J(p_i')|| \approx 0$

- Neighboring pixels have nearly similar disparity (surfaces in scene are roughly locally-uniform)
  e.g. $||d(p_i) - d(p_i + \Delta_{neighbor})|| \approx 0$

What is a good matching scheme under these assumptions?

Hint: think about HW4
Stereo as Energy Minimization

We formulate the matching problem with a cost function:

\[ E(d) = E_d(d) + \lambda E_s(d) \]

Data term \( E_d(d) \): Pixel/Window match

\[ E_d(d) = C(x, y, d(x, y)) \]

Smoothness term: \( E_s(d) \)

\[ C(x, y, d); \text{ the disparity space image (DSI)} \]

[Snavely]
Stereo as Energy Minimization

Simple pixel / window matching: choose the minimum of each column in the DSI independently:

$$d(x, y) = \arg\min_{d'} C(x, y, d')$$
Stereo as Energy Minimization

\[ E(d) = E_d(d) + \lambda E_s(d) \]

match cost:
\[ E_d(d) = \sum_{(x,y) \in I} C(x, y, d(x, y)) \]

smoothness cost:
\[ E_s(d) = \sum_{(p,q) \in \mathcal{E}} V(d_p, d_q) \]

\( \mathcal{E} \) : set of neighboring pixels

4-connected neighborhood

8-connected neighborhood

[Snavely]
Stereo as Energy Minimization

Find a smooth path through the DSI

Can you think of a way to do it?

Hint: Standard method in CS...

[Snavely]
Dynamic programming: Find minimal next step on the scan line:

\[
D(m, n, M) = \min(D(m-1, n-1, M), D(m-1, n, L), D(m-1, n-1, R)) + C'(m, n)
\]

\[
D(m, n, L) = \min(D(m-1, n-1, M), D(m-1, n, L)) + O
\]

\[
D(m, n, R) = \min(D(m, n-1, M), D(m, n-1, R)) + O,
\]

\( M \) - match,

\( L, R \) - Left visible, Right visible

An “Occlusion” loss \( O \) where no match is seen (L,R cases).
Results of Dynamic Programming approach

Problem: Streaking.

How should we fix it?
Stereo as Energy Minimization

Treat stereo as a labeling problem:

\[ L(p_i) = d_i \]

For each pixel \( p_i \) assign a depth label \( d_i \).

⇒ We can use the graph-cut method from segmentation!

In the 2D binary case (HW4) - use max-flow.

In the N-D case - no P algorithm, must be an approximation.
Continuous 3D Label Stereo Matching using Local Expansion Moves

Taniai et al., TPAMI 2017
A couple of good surveys:


A very good benchmark / competition system:

http://vision.middlebury.edu/stereo/eval3/
Questions?
Camera Calibration
Camera Parameters

Intrinsics $K$: $f$, $c_x$, $c_y$ (a 3x3 or 3x4 matrix, 3, 4 or 5 DOF)

Extrinsics: $[R \mid t]$ (a 3x4 or 4x4 matrix, 6 DOF)

Projection matrix: putting $K[R \mid t]$ together

\[
\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_x & \gamma & \delta_x & 0 \\ 0 & \phi_y & \delta_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_x \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_y \\ \omega_{31} & \omega_{32} & \omega_{33} & \tau_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}
\]
Extrinsic and Intrinsic At Once

The projection matrix is compound:

⇒ Need to find both extrinsic and intrinsic at once!

We do it in alternating steps:

- Fix **intrinsics** - find **extrinsics**
- Fix **extrinsics** - find **intrinsics**
Extrinsic calculation

Assume we have a good guess for intrinsic (for example \( f = 500, c_x = W/2, c_y = H/2 \))

Multiply by \( K^{-1} \) on both sides (“normalize” 2D points to the normalized camera):

\[
\lambda_i \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_x \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_y \\ \omega_{31} & \omega_{32} & \omega_{33} & \tau_z \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ w_i \\ 1 \end{bmatrix}
\]

\((x'_i, y'_i)\) in the \([-1, 1]\) range.

Problem: How will we know the 3D points??

Now we can solve directly a homogeneous linear system by spelling out:

\[
\begin{bmatrix} (\omega_{31}u_i + \omega_{32}v_i + \omega_{33}w_i + \tau_z)x'_i \\ (\omega_{31}u_i + \omega_{32}v_i + \omega_{33}w_i + \tau_z)y'_i \end{bmatrix} = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_x \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_y \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ w_i \\ 1 \end{bmatrix}
\]

A system of the type \( Ab = 0 \):

\[
\begin{bmatrix} u_1 & v_1 & w_1 & 1 & 0 & 0 & 0 & -u_1x'_1 & -v_1x'_1 & -w_1x'_1 & -x'_1 \\ 0 & 0 & 0 & u_4 & v_4 & w_4 & 1 & -u_4y'_1 & -v_4y'_1 & -w_4y'_1 & -y'_1 \\ u_2 & v_2 & w_2 & 1 & 0 & 0 & 0 & -u_2x'_2 & -v_2x'_2 & -w_2x'_2 & -x'_2 \\ 0 & 0 & 0 & u_2 & v_2 & w_2 & 1 & -u_2y'_2 & -v_2y'_2 & -w_2y'_2 & -y'_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_l & v_l & w_l & 1 & 0 & 0 & 0 & -u_lx'_l & -v_ly'_l & -w_ly'_l & -x'_l \\ 0 & 0 & 0 & u_l & v_l & w_l & 1 & -u_ly'_l & -v_ly'_l & -w_ly'_l & -y'_l \end{bmatrix} = 0
\]

Can use SVD to find the null-space: \( A = ULV^T \) ... [Prince]
Calibration Pattern

Need to make assumption on 3D - use a known pattern - a calibration pattern.

If we know the size: width, height (in mm) of the squares - we have a lot of 3D points to work with.

Finding them in the 2D image - use a corner detector, and try to connect them in lines.

Presto - we have 2D & 3D matching points.

Notice: The 3D points lie on a plane -
What do we know about planar projections?
Calibration Pattern

We can treat the planar projection as a **Homography**

-> Solve a more constrained problem, with less DOF (5, not 6)
Intrinsic Calibration

Compensate for extrinsic:

\[ \lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_x & \gamma & \delta_x & 0 \\ 0 & \phi_y & \delta_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} \]

Set up a similar linear system

\[ A_i = \begin{bmatrix} \frac{w_{11} u_i + w_{12} v_i + w_{13} w_i + \tau_x}{w_{31} u_i + w_{32} v_i + w_{33} w_i + \tau_z} & \frac{w_{21} u_i + w_{22} v_i + w_{23} w_i + \tau_y}{w_{31} u_i + w_{32} v_i + w_{33} w_i + \tau_z} & 1 \\ \frac{w_{31} u_i + w_{32} v_i + w_{33} w_i + \tau_z}{w_{31} u_i + w_{32} v_i + w_{33} w_i + \tau_z} & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \]

For \( h = [\phi_x, \gamma, \delta_x, \phi_y, \delta_y]^T \)
Questions?
Wrap Up

What we learned today:

- Basic stereo reconstruction pipeline
- Sliding window methods for stereo matching
- Global stereo matching constraints
- Camera calibration

Next time:

- Structured light
- Structure-from-Motion basics
- HW5 !! 😁