Talk Announcement

Luc Van Gool (KU Leuven) 25.02., 11:00h

Modeling Cities, Past and Present Dense

There is a quickly growing interest in the 3D modelling of cities, both existing and lost. In this talk some recent methods are discussed, that are intended to support such massive 3D modelling exercises. Several aspects are highlighted, depending whether the focus lies on the precision of 3D measurements (as in digital surveying), or the realism of visualisation. Underlying our strategies is a procedural modeling approach, which yields very compact but semantically enriched descriptions of buildings, for specific architectural styles. Attention is also paid to scalable data capture for these different applications. For instance, for digital surveying, the capture of 3D data along hundreds of km of road is necessary. We have designed a camera-equipped van with the necessary software to do so. In the case of modeling monuments - that require special attention given their rather complicated shapes - extra images are gathered automatically, without even having to specify the name or the existence of these monuments. Also, information about them is looked up automatically, e.g. to find out about the corresponding architectural style. As example case for the modelling of large-scale virtual cities, the Rome Reborn 2.0 project will be discussed, that has produced a model of the entire ancient city of Rome as it was around the 5th century AD.

Course Outline

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking
  - Motion and Optical Flow
  - Tracking with Linear Dynamic Models
  - Articulated Tracking & Novel Developments
- Repetition

Recap: Estimating Optical Flow

- Given two subsequent frames, estimate the apparent motion field \( u(x,y) \) and \( v(x,y) \) between them.
- Key assumptions:
  - Brightness constancy: projection of the same point looks the same in every frame.
  - Small motion: points do not move very far.
  - Spatial coherence: points move like their neighbors.

Solving the Aperture Problem

- Use all pixels in a \( K \times K \) window to get more equations.
- Least squares problem:
  \[
  \begin{bmatrix}
  I_u(p_1) & I_v(p_1) \\
  I_u(p_2) & I_v(p_2) \\
  I_u(p_{25}) & I_v(p_{25})
  \end{bmatrix}
  \begin{bmatrix}
  u \\
  v
  \end{bmatrix}
  = \begin{bmatrix}
  I_u(p_1) \\
  I_v(p_2) \\
  I_u(p_{25})
  \end{bmatrix}
  \]

- Minimum least squares solution given by solution of
  \[
  \begin{bmatrix}
  A^T A & 0 \\
  0 & 1
  \end{bmatrix}
  \begin{bmatrix}
  d \\
  h
  \end{bmatrix}
  =
  \begin{bmatrix}
  0 \\
  1
  \end{bmatrix}
  \]

- Recall the Harris detector!
Recap: Iterative Refinement

- Estimate velocity at each pixel using one iteration of LK estimation.
- Warp one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.
- Iterative procedure
  - Results in subpixel accurate localization.
  - Converges for small displacements.

Slide adapted from Steve Seitz

Recap: Coarse-to-fine Estimation

- Warps one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.

Slide adapted from Steve Seitz

Recap: Coarse-to-fine Estimation

- Run iterative L-K
- Warp & upsample
- Run iterative L-K
- ... 

Gaussian pyramid of image 1
Gaussian pyramid of image 2

Slide credit: Steve Seitz

Recap: Shi-Tomasi Feature Tracker

- Find good features using eigenvalues of second-moment matrix
  - Key idea: “good” features to track are the ones that can be tracked reliably.
- From frame to frame, track with Lucas-Kanade and a pure translation model.
  - More robust for small displacements, can be estimated from smaller neighborhoods.
- Check consistency of tracks by affine registration to the first observed instance of the feature.
  - Affine model is more accurate for larger displacements.
  - Comparing to the first frame helps to minimize drift.


Slide credit: Svetlana Lazebnik

Topics of This Lecture

- Tracking with Dynamics
  - Detection vs. Tracking
  - Tracking as probabilistic inference
  - Prediction and Correction
- Linear Dynamic Models
- The Kalman Filter
  - Kalman filter for 1D state
  - General Kalman filter
  - Limitations

Slide adapted from Steve Seitz

Detection vs. Tracking

- t=1
- t=2
- t=20
- t=21
Detection vs. Tracking

- **Detection**
  - We detect the object independently in each frame and can record its position over time, e.g., based on blob’s centroid or detection window coordinates.

- **Tracking with dynamics**
  - We use image measurements to estimate the object position, but also incorporate the position predicted by dynamics, i.e., our expectation of the object’s motion pattern.

Tracking with Dynamics

- **Key idea**
  - Given a model of expected motion, predict where objects will occur in next frame, even before seeing the image.

- **Goals**
  - Restrict search for the object
  - Improved estimates since measurement noise is reduced by trajectory smoothness.

- **Assumption: continuous motion patterns**
  - Camera is not moving instantly to new viewpoint.
  - Objects do not disappear and reappear in different places.
  - Gradual change in pose between camera and scene.

General Model for Tracking

- The moving object of interest is characterized by an underlying state $X$.
- State $X$ gives rise to *measurements* or *observations* $Y$.
- At each time $t$, the state changes to $X_t$ and we get a new observation $Y_t$.

State vs. Observation

- **Hidden state**: parameters of interest
- **Measurement**: what we get to directly observe

Tracking as Inference

- The hidden state consists of the true parameters we care about, denoted $X$.
- The measurement is our noisy observation that results from the underlying state, denoted $Y$.
- At each time step, state changes (from $X_{t-1}$ to $X_t$) and we get a new observation $Y_t$.
- Our goal: recover most likely state $X_t$ given
  - All observations seen so far.
  - Knowledge about dynamics of state transitions.
Steps of Tracking

- **Prediction**: What is the next state of the object given past measurements?
  \[ P(X_t | Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1}) \]

- **Correction**: Compute an updated estimate of the state from prediction and measurements.
  \[ P(X_t | Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1}, Y_t = y_t) \]

- Tracking can be seen as the process of propagating the posterior distribution of state given measurements across time.

Tracking as Induction

- **Base case**:
  - Assume we have an initial prior that predicts state in absence of any evidence: \( P(X_0) \)
  - At the first frame, correct this given the value of \( Y_0 \):
    \[ P(X_0 | Y_0 = y_0) = \frac{P(Y_0 | X_0) P(X_0)}{P(Y_0)} = \frac{P(Y_0 | X_0) P(X_0)}{P(Y_0)} \]

Simplifying Assumptions

- Only the immediate past matters

\[ P(X_t | Y_0, \ldots, Y_{t-1}) = P(X_t | X_{t-1}) \]

- Measurements depend only on the current state

\[ P(Y_t | X_t, Y_0, \ldots, Y_{t-1}) = P(Y_t | X_t) \]

Tracking as Induction

- **Base case**:
  - Assume we have initial prior that predicts state in absence of any evidence: \( P(X_0) \)
  - At the first frame, correct this given the value of \( Y_0 \):
  - Given corrected estimate for frame \( t \):
    - Predict for frame \( t+1 \)
    - Correct for frame \( t+1 \)

Induction Step: Prediction

- Prediction involves representing \( P(X_t | y_0, \ldots, y_{t-1}) \) given \( P(X_{t-1} | y_0, \ldots, y_{t-2}) \)

\[ P(X_t | y_0, \ldots, y_{t-1}) = \int P(X_t | X_{t-1}, y_0, \ldots, y_{t-1}) dX_{t-1} \]

- Law of total probability
  \[ P(A) = \int P(A | B) dB \]

Induction Step: Prediction

- Prediction involves representing \( P(X_t | y_0, \ldots, y_{t-1}) \) given \( P(X_{t-1} | y_0, \ldots, y_{t-2}) \)

\[ P(X_t | Y_0, \ldots, Y_{t-1}) = \int P(X_t | X_{t-1}, Y_0, \ldots, Y_{t-1}) dX_{t-1} = \int P(X_t | X_{t-1}, Y_0, \ldots, Y_{t-1}) P(X_{t-1} | Y_0, \ldots, Y_{t-1}) dX_{t-1} \]

- Conditioning on \( X_t \):
  \[ P(A | B) = \frac{P(A | B) P(B)}{P(B)} \]
Induction Step: Prediction

- Prediction involves representing $P(X_t | y_0, \ldots, y_{t-1})$
given $P(X_{t-1} | y_0, \ldots, y_{t-1})$

\[ P(X_t | y_0, \ldots, y_{t-1}) = \int P(X_t | X_{t-1}, y_0, \ldots, y_{t-1}) P(X_{t-1} | y_0, \ldots, y_{t-1}) dX_{t-1} \]

- Induction Step: Prediction

\[ P(X_t | y_0, \ldots, y_{t-1}) = \int P(X_t | X_{t-1}, y_0, \ldots, y_{t-1}) P(X_{t-1} | y_0, \ldots, y_{t-1}) dX_{t-1} \]

Induction Step: Correction

- Correction involves computing $P(X_t | y_0, \ldots, y_{t-1})$
given predicted value $P(X_{t-1} | y_0, \ldots, y_{t-1})$

\[ P(X_t | y_0, \ldots, y_{t-1}) = \frac{P(y_t | X_t, y_0, \ldots, y_{t-1}) P(X_t | y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})} \]

Bayes rule

\[ P(A | B) = \frac{P(B | A) P(A)}{P(B)} \]

Summary: Prediction and Correction

- Prediction:

\[ P(X_t | y_0, \ldots, y_{t-1}) = \int P(X_t | X_{t-1}) P(X_{t-1} | y_0, \ldots, y_{t-1}) dX_{t-1} \]

- Correction:

\[ P(X_t | y_0, \ldots, y_{t-1}) = \int P(y_t | X_t) P(X_t | y_0, \ldots, y_{t-1}) dX_t \]
Topics of This Lecture

- Tracking with Dynamics
  - Detection vs. Tracking
  - Tracking as probabilistic inference
  - Prediction and Correction
- Linear Dynamic Models
  - The Kalman Filter
    - Kalman filter for 1D state
    - General Kalman filter
    - Limitations

Notation Reminder

- $x \sim N(\mu, \Sigma)$
  - Random variable with Gaussian probability distribution that has the mean vector $\mu$ and covariance matrix $\Sigma$.
  - $x$ and $\mu$ are $d$-dimensional, $\Sigma$ is $d \times d$. If $x$ is 1D, we just have one $\Sigma$ parameter: the variance $\sigma^2$

Linear Dynamic Models

- Dynamics model
  - State undergoes linear transformation $D_t$ plus Gaussian noise
    \[
    x_0 \sim N\left(D_0 x_0 + \Sigma_d\right)
    \]
- Observation model
  - Measurement is linearly transformed state plus Gaussian noise
    \[
    y_0 \sim N\left(M y_0 + \Sigma_m\right)
    \]

Example: Randomly Drifting Points

- Consider a stationary object, with state as position.
  - Position is constant, only motion due to random noise term.
  \[
  x_i = p_i + \Delta
  \]
  \[
  p_i = p_{i-1} + \xi
  \]
  \[
  \Rightarrow \text{State evolution is described by identity matrix } D=I
  \]
  \[
  x_i = D x_{i-1} + \text{noise} = I p_{i-1} + \text{noise}
  \]

Example: Constant Velocity (1D Points)

- State vector: position $p$ and velocity $v$
  \[
  x_i = \begin{bmatrix} p_i \\ v_i \end{bmatrix}
  \]
  \[
  p_i = \begin{bmatrix} p_{i-1} \\ v_{i-1} \end{bmatrix} + \xi
  \]
  \[
  \Rightarrow \text{Measurement is position only}
  \]
  \[
  y_i = M x_i + \text{noise}
  \]
Example: Constant Velocity (1D Points)
• State vector: position $p$ and velocity $v$
\[
\begin{align*}
x_t &= \begin{bmatrix} p_t \\ v_t \end{bmatrix} \\
p_{t+1} &= p_{t+1} + (\Delta t)v_{t+1} + \xi \\
v_{t+1} &= v_{t+1} + \zeta \\
x_t &= D_x x_{t-1} + \text{noise}
\end{align*}
\]
\[
\begin{bmatrix} p_t \\ v_t \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + \text{noise}
\]
• Measurement is position only
\[
y_t = M_x + \text{noise} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + \text{noise}
\]

Example: Constant Acceleration (1D Points)
• State vector: position $p$, velocity $v$, and acceleration $a$.
\[
\begin{align*}
x_t &= \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} \\
p_{t+1} &= p_{t+1} + (\Delta t)v_{t+1} + \xi \\
v_{t+1} &= v_{t+1} + \zeta \\
a_{t+1} &= a_{t+1} + \zeta \\
x_t &= D_x x_{t-1} + \text{noise}
\end{align*}
\]
\[
\begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \\ a_{t-1} \end{bmatrix} + \text{noise}
\]
• Measurement is position only
\[
y_t = M_x + \text{noise} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \\ a_{t-1} \end{bmatrix} + \text{noise}
\]

Example: General Motion Models
• Assuming we have differential equations for the motion
  E.g. for (undamped) periodic motion of a pendulum
  \[
d^2 p / dt^2 = -p
\]
• Substitute variables to transform this into linear system
\[
p_1 = p \\
p_2 = p' = dp/ dt \\
p_3 = p'' = d^2 p / dt^2
\]
• Then we have
\[
\begin{bmatrix} p_{t+1} \\ p_{t+2} \\ p_{t+3} \\ p_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & \Delta t & 0 \\ 0 & 0 & 1 & \Delta t \\ -1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t} \\ p_{t} \\ p_{t} \\ p_{t} \end{bmatrix} + \text{noise}
\]

Topics of This Lecture
• Tracking with Dynamics
  - Detection vs. Tracking
  - Tracking as probabilistic inference
  - Prediction and Correction
• Linear Dynamic Models
• The Kalman Filter
  - Kalman filter for 1D state
  - General Kalman filter
  - Limitations
The Kalman Filter

- Method for tracking linear dynamical models in Gaussian noise
- The predicted/corrected state distributions are Gaussian
  - You only need to maintain the mean and covariance.
  - The calculations are easy (all the integrals can be done in closed form).

**Kalman Filter for 1D State**

Want to represent and update

\[ P(x_t | y_0, \ldots, y_{t-1}) = N(\mu_t^*, (\sigma_t^*)^2) \]
\[ P(x_t | y_0, \ldots, y_t) = N(\mu_t^*, (\sigma_t^*)^2) \]

**1D Kalman Filter: Prediction**

- Have linear dynamic model defining predicted state evolution, with noise
  \[ X_t \sim N(dx_{t-1}, \sigma^2) \]
- Want to estimate predicted distribution for next state
  \[ P(X_t | y_0, \ldots, y_{t-1}) = N(\mu_t^*, (\sigma_t^*)^2) \]
- Update the mean:
  \[ \mu_t^* = d\mu_{t-1} \]
- Update the variance:
  \[ (\sigma_t^*)^2 = \sigma^2 + (d\sigma_{t-1}^2) \]

**1D Kalman Filter: Correction**

- Have linear model defining the mapping of state to measurements:
  \[ Y_t \sim N(mx_t, \sigma^2) \]
- Want to estimate corrected distribution given latest measurement:
  \[ P(X_t | y_0, \ldots, y_t) = N(\mu_t^*, (\sigma_t^*)^2) \]
- Update the mean:
  \[ \mu_t^* = \frac{\mu_t^* \sigma_m^2 + my_t (\sigma_t^*)^2}{\sigma_m^2 + m^2 (\sigma_t^*)^2} \]
- Update the variance:
  \[ (\sigma_t^*)^2 = \frac{\sigma_m^2 (\sigma_t^*)^2}{\sigma_m^2 + m^2 (\sigma_t^*)^2} \]
**Prediction vs. Correction**

\[ \mu_i = \frac{\mu_m^2 \sigma_m^2 + my_i (\sigma_y^2)^2}{\sigma_m^2 + m^2 (\sigma_y^2)^2} \]

\[ (\sigma_i^2) = \frac{\sigma_m^2 (\sigma_y^2)^2}{\sigma_m^2 + m^2 (\sigma_y^2)^2} \]

- What if there is no prediction uncertainty \((\sigma_m^2 = 0)\)?
  \[ \mu_i = \mu_m \]

  The measurement is ignored!

- What if there is no measurement uncertainty \((\sigma_y^2 = 0)\)?
  \[ \mu_i = \frac{y_i}{m} \]

  The prediction is ignored!

---

**Recall: Constant Velocity Example**

The state is 2D: position + velocity measurements. The figure shows the state over time with measurements before and after. The predicted mean estimate and corrected mean estimate are indicated with bars showing variance estimates before and after measurements.
Kalman Filter: General Case (>1dim)

• What if state vectors have more than one dimension?

\[ \begin{align*}
    x'_i &= D x_{i-1} \\
    \Sigma'_i &= D \Sigma_{i-1} D^T + \Sigma_i
\end{align*} \]

**PREDICT**

\[ \begin{align*}
    K_i &= \Sigma_i M_i \left( M_i \Sigma_i M_i^T + \Sigma_{m_i} \right)^{-1} \\
    x'_i &= x_i - K_i \left( y_i - M_i x'_i \right) \\
    \Sigma'_i &= (I - K_i M_i) \Sigma_i
\end{align*} \]

**CORRECT**

“Residual”

More weight on residual when measurement error covariance approaches 0.

Less weight on residual as a priori estimate error covariance approaches 0.

Summary: Kalman Filter

• **Pros:**
  - Gaussian densities everywhere
  - Simple updates, compact and efficient
  - Very established method, very well understood

• **Cons:**
  - Unimodal distribution, only single hypothesis
  - Restricted class of motions defined by linear model

Why Is This A Restriction?

• Many interesting cases don’t have linear dynamics
  - E.g. pedestrians walking

• E.g. a ball bouncing

Ball Example: What Goes Wrong Here?

• Assuming constant acceleration model

• Prediction is too far from true position to compensate...

• Possible solution: Extended Kalman Filter
  - Keeps multiple different motion models in parallel
  - I.e. would check for bouncing at each time step

Tracking - Difficulties

References and Further Reading

• A very good introduction to tracking with linear dynamic models and Kalman filters can be found in Chapter 17 of

  - D. Forsyth, J. Ponce,
    *Computer Vision - A Modern Approach.*
    Prentice Hall, 2003
4 Regeln zum korrekten Ausfüllen der Fragebögen


2) □ □ □ □ = Falsch
□ □ □ □ = Richtig

3) □ □ □ = Falsch
□ □ □ □ □ □ □ □ □ = Richtig: Bei Irrtum Kästchen schwarz und anders ankreuzen.

4) Handschriftliche Texte innerhalb und nicht zu dicht an den Rand der Begrenzungslinie schreiben.