Sensor and Actuator Networks

Vision based Positioning

Sen Wang
Department of Computer Science

February 2017
Vision based Positioning

- Flying Robot
- Self-Driving Car
- Space Robot
- Virtual Reality
Why Vision based Positioning

• camera: light-weight, low-cost, rich information
• flexible: mobile, flying, underwater robots
• pervasive: phones, cars, various applications
• more accurate than wheel odometry
• no wheel slip
Visual Odometry

Move to see, see to move

- motion: changes on the images
- changes: estimate motion, pose (x, y, z, roll, pitch, yaw)
Visual Odometry

- motion: changes on the images
- changes: estimate motion, pose (x, y, z, roll, pitch, yaw)

frame-to-frame motion for a video (a sequence of images)
Types

• Geometry based Methods: extract geometric constraints from imagery to estimate motion
  • Sparse feature based methods
  • Direct methods

• Learning based Methods (focus): use machine learning techniques without explicitly applying geometric theory
Sparse Feature based Methods
Sparse Feature based Methods

computes camera trajectory incrementally
Direct methods

• Direct methods: **find camera poses** that maximise the probability of observing current image (**pixel intensities**)

• assumption: **brightness constancy** (photometric calibration)
• minimum of the photometric cost function
Direct methods

Direct minimization of photometric error

\[ E(\xi) = \sum_{p_i \in \Omega_{\text{ref}}} (I_{\text{ref}}(p_i) - I(p_i'))^2 \]

- camera pose
- ref. image
- new image
- ref. depth

sum over valid ref. pixel

\[ p_i' = \omega(p_i, d, \xi) = \pi(K(R_{\xi}K^{-1}d) \begin{pmatrix} p_{i,x} \\ p_{i,y} \\ 1 \end{pmatrix} + t_{\xi}) \]

\[ \pi(x, y, z) := \begin{pmatrix} x/z \\ y/z \end{pmatrix} \]

\[ \begin{pmatrix} R_{\xi} & t_{\xi} \\ 0 & 1 \end{pmatrix} := \exp(\xi) \]

\[ \omega(p_i, d, \xi) \] "warps" a pixel from ref. image to new image

courtesy of Jakob Engel

minimise using the Gauss-Newton or LM algorithm
Good for VO?
## Comparison

<table>
<thead>
<tr>
<th>Indirect</th>
<th>Direct</th>
</tr>
</thead>
<tbody>
<tr>
<td>can only use (&amp; reconstruct) corners</td>
<td>can use (&amp; reconstruct) everything that has image gradient</td>
</tr>
<tr>
<td>faster</td>
<td>slower (but good for parallelism)</td>
</tr>
<tr>
<td>can deal well with geometric noise in the system (rolling shutter).</td>
<td>does not model geometric noise in the system (rolling shutter BAD).</td>
</tr>
<tr>
<td>Data association (KP detection &amp; matching) based on incomplete information.</td>
<td>No initial, fixed data association. Still need to determine visibility etc.</td>
</tr>
<tr>
<td>no need for good initialization.</td>
<td>needs good initialization (solved by IMU)</td>
</tr>
</tbody>
</table>

courtesy of Jakob Engel
Drift and Scale Problems

• drift problem of VO
  • errors accumulate over time
  • similar to wheel odometry

• scale problem of monocular VO
  • metric scale cannot be recovered, why?
Scale Problem of Monocular VO
Better Accuracy and Robustness

How?

• better camera
  • wide dynamic range, global shutter, high frame rate, wide-angle, etc.
• better algorithm
  • outliers: random sample consensus (RANSAC)
• drift problem
  • local optimisation
  • visual Simultaneous Localization and Mapping (SLAM)
• scale problem for monocular VO
  • stereo VO
  • other sensors, information (fixed height)
• visual inertial odometry
  • fuse with Inertial Measurement Units (IMUs): Google Project Tango
  • scale?
VO vs. Visual SLAM (1/2)

- The goal of SLAM in general is to obtain a global, consistent estimate of the robot path. This is done through identifying loop closures. When a loop closure is detected, this information is used to reduce the drift in both the map and camera path (global bundle adjustment).

- Conversely, VO aims at recovering the path incrementally, pose after pose, and potentially optimizing only over the last $m$ poses path (windowed bundle adjustment).

Image courtesy of Clemente et al. RSS’07
VO vs. Visual SLAM (2/2)

• VO only aims to the **local consistency** of the trajectory

• SLAM aims to the **global** consistency of the trajectory and of the map

• VO can be used as a **building block** of SLAM

• VO is SLAM before closing the loop!

• The choice between VO and V-SLAM depends on the tradeoff between performance and consistency, and simplicity in implementation.

• VO trades off consistency for real-time performance, without the need to keep track of all the previous history of the camera.
More on Geometry based VO/SLAM

• books
• papers sent to you
  • Visual Odometry Tutorial: Part I and II
• open-source algorithms to try
  • LIBVISO2
  • MonoSLAM
  • PTAM
  • ORB-SLAM

• LSD-SLAM
• SVO: Semi-direct Visual Odometry
• DSO: Direct Sparse Odometry

• RGB-D SLAM
• RTAB-Map
• ..........
Video/Demo

current vision based localisation systems
sparse feature method
dense method
Types

• Geometry based Methods: extract geometric constraints from imagery to estimate motion
  • Sparse feature based methods
  • Direct methods

• Learning based Methods
  • Traditional learning based methods
  • Deep Learning based methods

several decades
excellent accuracy
very popular

several years
rare
Learning based Methods

• Learning based methods aim to derive motion model and infer VO from sensor data by using ML techniques without explicitly applying geometric theory.
  • Traditional learning based methods
  • Deep Learning based methods
Traditional learning based methods

- Sparse optical flow
  - K Nearest Neighbour (KNN)
  - Gaussian Processes (GP)
  - Support Vector Machines (SVM) regression

- difficult to use raw images
  - high-dimensional
  - redundant information
Why Deep Learning

- curse of data
  - high-dimensional: raw images
  - big: sensor data from hundreds of thousands of robots/devices
Hand-Crafted Features by Human

Pervasive Data
- Time-series data
- Vision
- Point cloud

Feature Extraction (hand-crafted)

Inference
- Activities, Context, ...
- Objects, Scene types, ...
- Objects, Structure, Semantics, ...
End-to-End Learning

Pervasive Data

- time-series data
- vision
- point cloud

End-to-End Learning

automatically learn most effective feature representation to solve the problem

Inference

- Activities, Context, ...
- Objects, Scene types, ...
- Structure, Semantics, ...
Deep Learning

Training (supervised learning)

- big data
- forward: prediction
- backward: error
- data label

Testing

- test data
- forward: prediction
- trained DNN
Convolutional Neural Network - CNN

• recap previous courses
• similar to feed-forward Neural Networks

• Merits:
  • more efficient to implement
  • vastly reduce the amount of parameters

• Important Ideas:
  • local connectivity
  • parameter sharing

• Main Type of Layers:
  • convolutional layer
  • activation layer
  • pooling layer
CNN Layers

- **Local Connectivity**
- **Parameter Sharing**

courtesy of Andrej Karpathy
Recurrent Neural Network - RNN

- temporal processing and learn sequences
- sequential data: voice, text, language, video, sensor data, etc.
- flexible configurations: one to one, one to many, many to many
- **memory**

video captioning, translation: a sequence of input maps to a sequence of output maps

courtesy of Andrej Karpathy
Recurrent Neural Network - RNN

- loops along time: memory to save information
- unfold over time
- dependence and connection
- Backpropagation Through Time (BPTT)

Image courtesy of Christopher Olah
Simple Recurrent Neural Networks

• a sequence of $x$ at every time step

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

Image courtesy of Christopher Olah
Example: Language Model

- vocabulary: [h,e,l,o]
- input: one-hot encoding

\[ h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t) \]

courtesy of Andrej Karpathy
Long Short-Term Memory - LSTM

- LSTM to avoid long-term dependency problem
- gates: control information flow, cell states
- forget gate: “remember” or “forget” states

![Diagram of LSTM network](image courtesy of Christopher Olah)
LSTM Cell: Step by Step

\[
\begin{align*}
i_k &= \sigma(W_i I_k + W_h h_{k-1} + b_i) \\
f_k &= \sigma(W_f I_k + W_h h_{k-1} + b_f) \\
g_k &= \tanh(W_g I_k + W_h h_{k-1} + b_g) \\
c_k &= f_k \odot c_{k-1} + i_k \odot g_k \\
o_k &= \sigma(W_o I_k + W_h h_{k-1} + b_o) \\
h_k &= o_k \odot \tanh(c_k)
\end{align*}
\]
Bidirectional-LSTM

- forward and backward layers
- future information is reachable from current state
- output can access information from past and future states
LSTM Variants

Gated Recurrent Units (GRUs)


eight LSTM variants

1. No Input Gate (NIG)
2. No Forget Gate (NFG)
3. No Output Gate (NOG)
4. No Input Activation Function (NIAF)
5. No Output Activation Function (NOAF)
6. No Peepholes (NP)
7. Coupled Input and Forget Gate (CIFG)
8. Full Gate Recurrence (FGR)

\[
\begin{align*}
    z &= \text{sigmoid}(W_{xz}x_t + b_z) \\
    r &= \text{sigmoid}(W_{xr}x_t + W_{hr}h_t + b_r) \\
    h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + \tanh(x_t) + b_h) \odot z \\
    &\quad + h_t \odot (1 - z)
\end{align*}
\]

MUT1:

\[
\begin{align*}
    z &= \text{sigmoid}(W_{xz}x_t + W_{hz}h_t + b_z) \\
    r &= \text{sigmoid}(x_t + W_{hr}h_t + b_r) \\
    h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + W_{zh}x_t + b_h) \odot z \\
    &\quad + h_t \odot (1 - z)
\end{align*}
\]

MUT2:

\[
\begin{align*}
    z &= \text{sigmoid}(W_{xz}x_t + W_{hz} \tanh(h_t) + b_z) \\
    r &= \text{sigmoid}(W_{xr}x_t + W_{hr}h_t + b_r) \\
    h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + W_{zh}x_t + b_h) \odot z \\
    &\quad + h_t \odot (1 - z)
\end{align*}
\]

MUT3:
Deep RNN

• multiple layers: high level features

• more complex connections between inputs and features
End-to-End Visual Odometry
Architecture

• achieve monocular Visual Odometry (VO) in an end-to-end, sequence-to-sequence manner based on Deep Learning, i.e., directly estimating poses from a sequence of raw RGB images

• leverage a large number of images

• feature extraction + sequential learning
CNN based Feature Extraction

- Convolutional Neural Network (CNN)
- 2 consecutive images as input
- feature extraction on raw RGB images
- generalise to new environments, NOT confined to trained ones
- geometric feature representation
RNN based Sequential Learning

• sequential dependence and motion dynamics of an image sequence are automatically learnt by Recurrent Neural Network (RNN)

• convolutional feature is passed to RNN for sequential modelling

• LSTM as RNN

• multiple layers of LSTM as deep structure
Novel SE(3) Layer

- Special Euclidean Group SE(3) for transformation
- pose composition: matrix multiplication in the context of SE(3)
- no hyper-parameter to be learned
- direct pose feedback

\[
T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}
\]

\[
T_{WC_k} = T_{WC_{k-1}} T_{C_{k-1} C_k}
\]
Experiments

• Testing scenarios: outdoor driving, indoor Micro Aerial Vehicle, indoor pedestrian, outdoor fixed-wing UAV

• Comparison (VO version):
  - VISO2: sparse feature based VO
  - ORB-SLAM: state-of-the-art sparse feature based visual SLAM
  - LSD-SLAM: state-of-the-art direct method based visual SLAM

• Scale recovery:
  - VISO2: fixed camera height
  - ORB-SLAM and LSD-SLAM: similarity transformation to ground truth
  - Deep-VO: automatic
Outdoor Driving

• KITTI VO/SLAM Benchmark and Raw Dataset
• Trained on
  - 5 of VO training sequences: test on other training sequences
  - all training sequences: test on testing and raw sequences
• Results:
  - monocular VISO2: big drift
  - ORB-SLAM: scale drift without loop closure detection
  - LSD-SLAM: lose tracking (low frame rate and high speed)
Indoor Flying

- ETH EuRoC dataset: indoor Micro Aerial Vehicle
- quadrotoor in an indoor large machine hall
- trained on 4 sequences of Machine Hall and tested on another one
Indoor Motion: Test in Office Building

- **Training data:** Wolfson Building of Department of Computer Science
- **Test:**
  - Robert Hooke Building of Department of Computer Science
  - Natural History Museum
- **Geometry based methods**
  - prone to lose tracking
  - texture-less: corridor, white wall
  - agile motion: fast turn

---

ORB-SLAM: lose tracking

Deep-VO

Sensor and Actuator Networks - AIMS CDT - Sensor Networks Group
Indoor Motion: Test in Museum

- Natural History Museum
  - Challenging lighting conditions
- Café with walking people
- Deep-VO: see motion in video

ORB-SLAM: scale problem

LSD-SLAM: lose tracking
Supermarket

Demo
Deep Learning based Visual Odometry

- field demo/test for Microsoft Research (Redmond) in US

Deep Learning based VIO

• Visual Inertial Odometry: visual and inertial sensors
• multi-rate Recurrent Neural Network
• robust to
  • extrinsic calibration errors
  • time synchronisation errors

End-to-End Global Localisation
Global Localisation

• global localisation:
  • not only place recognition
  • 6 DoF pose in the environments, maps

• loop closure detection for SLAM

• re-localisation: recovery after losing tracking
Perceptual Aliasing

- different locations have similar appearance
- same location has different appearances
- how to distinguish, then localise?

Images from RobotCar Dataset
Methods
End-to-End Global Localisation

- learn most effective features
- spatial and temporal models
- a sequence of images (video)

[CVPR'17] “6-DoF video-clip re-localisation.”
Image Features: CNN

- images -> features
- extract features by CNN
  - AlexNet
  - VGGNet
  - GoogLeNet
  - ResNet
  - .......
Temporal Modelling: Bidirectional RNN

- a sequence of image features -> temporal model
- video clip:
  - dynamics, motion
  - robot, car, pedestrian, etc.
- bi-directional LSTM
  - connection between steps

![Diagram of Temporal Modelling: Bidirectional RNN]
Global Poses

• a pose for each image -> sequence
• DNN model
  • “compressed map”
  • feature matching
  • loop closure detection
  • pose estimation
Test on RobotCar Dataset

without
with
temporal modelling
Practical

• AR Drone: front camera -> images
• Vicon motion capture system: GT
• Robot Operating System (ROS)
  • control AR Drone
  • log data: images, GT
• design/train models for localisation
• compare models, know why
• use estimated poses for applications
Pros and Cons?

geometry based methods

learning based methods