TSBB15 Computer Vision

Lecture 7

Object tracking project

Project web page

- More information about the project can be found here:
- www.cvl.isy.liu.se/education/undergraduate/tsbb15/object-tracking-project
  - Datasets
  - References
  - Code
  - …

Object tracking

Technical goals of this project:

- Given an image sequence,
  detect and **track** all moving objects
- Indicate objects with **boxes** + **identity numbers**

Assumptions:

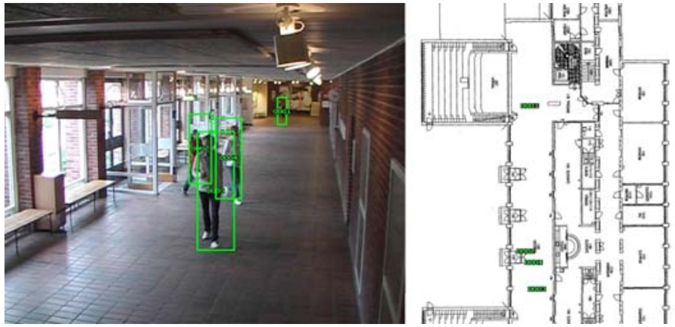
- Static camera
- Static background
- Humans or cars are moving on a flat ground plane

Object tracking

- Large application area:
  - surveillance, traffic, …
- Many spin-offs
  - http://www.cognimatics.com
- Several research projects, e.g.,
  - FOI
  - CAVIAR
  - PETS
- Student projects (e.g. Puck-O-Vision)

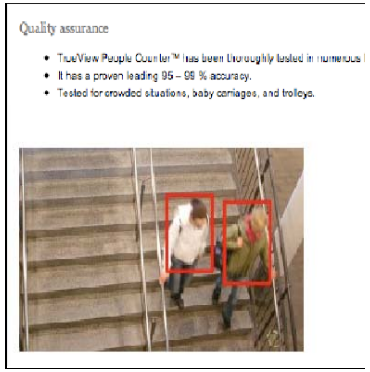
Object tracking

Example



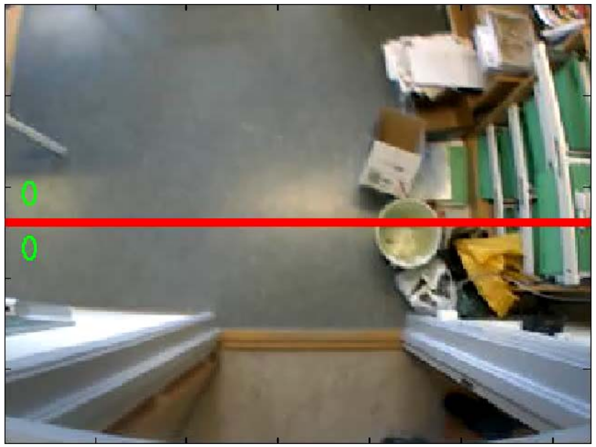
From PhD Thesis of Håkan Ardö, LTH

Object tracking

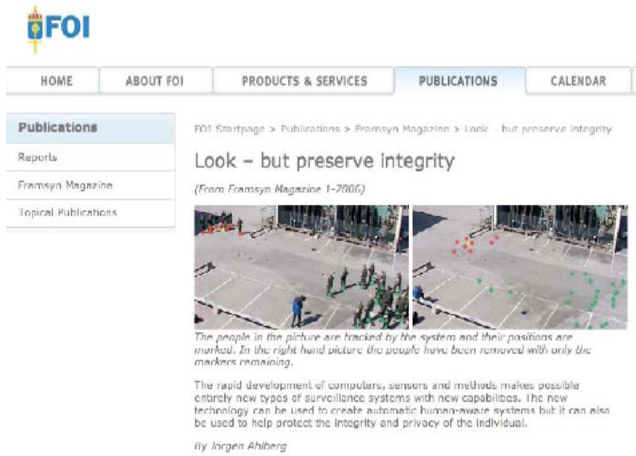


TrueView from Cognimatics

People counting

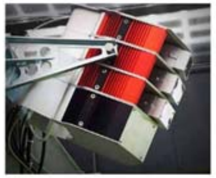


Object tracking



Sports

Example:
Tracab system for
automatic tracking of
football players.
From CyronHego



TRACAB is the only system proven to deliver true three-dimensional tracking in a sports arena in REAL real-time.

Challenges

- Shadows / reflections from moving objects
  - Create “ghost” objects
- Uneven illumination of the scene
  - Visual appearance of an object changes
- 3D rotations
  - Object appearance changes
- Occlusion
  - Ambiguities in object identity

Toolbox of methods

- Since we have a static camera & background:
  - Use *background modeling* to detect motion
- To robustly determine object identity:
  - KL-tracking
  - Prediction
  - Ground plane mapping + 3D tracking
- Shadows/reflections can be managed by:
  - Color modeling

Background modeling

A popular application of mixture models

- Described in lecture 6
- Estimate a mixture model for the image background in each pixel
  - o 1 model sometimes sufficient
  - o 2 or more models are sometimes used to manage minor fluctuations
- Pixel values far from the mixture model are seen as foreground pixels
- Popular way to track e.g. people and cars in stationary surveillance cameras
- Fast compared to dense motion estimation

Background modeling

- Master thesis project at CVL, LiU, by John Wood
- Background modeling + shadow detection

Background modeling

Two proposed approaches:

- Gaussian Mixture models,
  - described in lecture 6
  - Also pseudo-code in J. Wood's master thesis, section 4.3
- Median filtering
  - Masters thesis of J. Wood, section 2.2
  - PhD thesis of H. Ardö, section 3.2.2

Foreground segmentation

- Background modeling gives a likelihood image with $p(I(x,y)|\text{background})$
  - Lecture 6
- From this probability, find **consistent regions** by
  - thresholding
  - morphological shrink + expand for cleanup
  - labeling
- From these regions, extract **bounding boxes**
  - The smallest rectangle that includes the detected object

Object identity

- Start with detecting moving objects in each frame
- Next: determine which object in frame t corresponds to which one in frame $t + 1$
- Simple approach:
  - Check how much all pairs of objects overlap
  - Connect pairs of **maximal overlap**
- More robust alternative:
  - Use temporal filtering for **predicting** position of boxes
  - Check overlap of predicted boxes

Temporal filtering

- Temporal filtering of object locations e.g. IIR or Kalman
  - From temporal filtering we can predict where the object will end up in the next frame
- Can help to maintain object identity in case of
  - occlusion
  - close objects, moving fast
  - or noise in general

Ground plane modeling

- We assume that the objects are moving around on a planar surface, the ground plane:
  - There is a homography transformation between image position and position in the plane (TSBB06)
  - This homography can be estimated (TSBB06)
  - "foot points" of object can be mapped to a position in the ground plane
  - The ground plane position can be temporally filtered

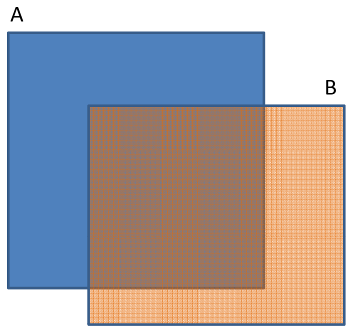
Datasets

- The scientific community has produced a number of standard dataset for motion analysis:
  - CAVIAR
  - PETS
  - IVSS
  - changedetection.net
- They typically contain:
  - Video sequences (MPEG + JPEG)
  - Annotation (labeling of objects) in some datasets

Performance evaluation

- The datasets have ground truth (GT)
- Compare your boxes with GT
- Possible performance measures
  - Average overlap error
  - Number of changes of identity for an object
  - ...

Overlap



overlap = $\frac{\mathrm{Area}(A \cap B)}{\mathrm{Area}(A \cup B)}$

$0 \leq \text{overlap} \leq 1$

overlap error = $1 - \frac{\mathrm{Area}(A \cap B)}{\mathrm{Area}(A \cup B)}$

Examples from CAVIAR

Camera 1: Wide field of view in a large room

Camera 2: Surveillance in a shopping mall

Guidance

- Each project group is assigned a PhD student that gives guidance
- Each group can expect to use 2h of guidance time per week
- Mainly at times agreed upon by both parties
- Guidance implies
  - Give advice
  - Give references to literature, code, and data

Subversion

- Each project group will have access to a Subversion (SVN) based repository
  - Individual check-in
- Use the repository to store:
  - all external code used in the project
  - all code produced in the project
  - all documents produced in the project
- See “Subversion pitfalls” on course web page

Data storage

- Each project group can get a separate 5 GB file area
  - No backup
  - Deleted after project is finished
  - If needed: ask your project guide

Cooperation and plagiarism

- Discussions between students and between project groups ***is allowed***
- Each group must declare which code/text has been produced by the group and which is external
- External material (e.g. code) must be approved
  - Allowed code packages are listed on web page
  - Ask the examiner **before using other packages**
- Put ”own” and ”external” code in different directories
- Material declared as “own” must not be plagiarized from external sources

Design plan

- Each project must produce a design plan
  - Describes the important functionalities (blocks)
  - Describes data flow between the blocks
  - Describes interfaces
  - Describes methods of implementations
- The design plan should also describe who is the principal implementer of the block
  - ***Each student must be responsible for the design of at least one major block***
  - Should be able to answer questions on that block during the final seminar

Project context

- Use reasonable project management
  - make a design plan
  - discuss it with your guide
  - get it approved
  - divide work among project members
    - Each member must have an assigned part of the technical work
    - Presents that part at the seminar
    - Answers questions by the examiner (and audience) on that part

Project context

- Produce test cases
  - Should be simple, easy to find expected result
  - Test early implementations on these
- Deliverables
  - A good presentation
    - Including examination of each project member
  - A report

Testing

- **Testing is one of the most important parts of the project**
- Divide your system into subparts that can be easily tested
- You should test using *test data*
  - Produces obvious or specified behavior
  - Usually not the same data as the final system should operate on
  - Can be synthetic data generated to produce well-known output from your system
- Do not integrate subparts into a larger system until the subparts are properly tested

Deliverables

- A good presentation
  - Including examination of each project member
- A written project report
  - Which problem is solved?
  - How is it solved?
  - What is the result? Performance evaluation!
  - Why did you get this result?
  - References
  - Targeted to your fellow students, not doing this project (explain what you are doing!)

Time table 2016

- **Feb 3:** formation of project groups
  - Problem analysis
  - Specification of required functionalities
  - Selection of methods
  - Who does what
  - First contact with project guide
- **Feb 12:** approval of design plan
- **March 22:** report handed in to guide
- **April 5:** presentation of project results
- Each student contributes $\approx$80h (3hp) of work/project

Implementation

The project can be implemented in

- Matlab
  - real-time processing difficult
  - no real-time requirements
- C/C++
  - close to real-time should be possible
  - but is NOT a requirement
- Several code packages exist
  - OpenCV for C/C++
  - See list on web page
  - External software for background modeling not allowed

OpenCV

- Version 2.4.2
  - Installed in ISY computer rooms
- Contains lots of useful functions:
  - Image / video / camera input
  - Image display
  - KLT-tracker
  - Harris-operator
  - Morphology, statistics
- Requires C/C++ skills
- See "OpenCV pitfalls" on course web page
- Use Matlab otherwise (has similar stuff)

Required functionalities

- For all groups:
  - Tracking of objects in a simple sequence
  - Management of object identity
  - Evaluation of results
    - E.g., accuracy relative to ground truth
- Optional functionality:
  - Management of shadows
  - Management of spurious background motions
  - Occlusion management, e.g., prediction or ground plane modeling
- (Groups of 3 students: 1 optional function)
- Groups of 4 students: 2 optional functions
- Groups of 5 students: 3 optional functions