

3D Computer Vision

Radim Šára Martin Matoušek

Center for Machine Perception
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague

<https://cw.felk.cvut.cz/doku.php/courses/a4m33tdv/>

<http://cmp.felk.cvut.cz>

<mailto:sara@cmp.felk.cvut.cz>

phone ext. 7203

rev. September 18, 2012



Open Informatics Master's Course

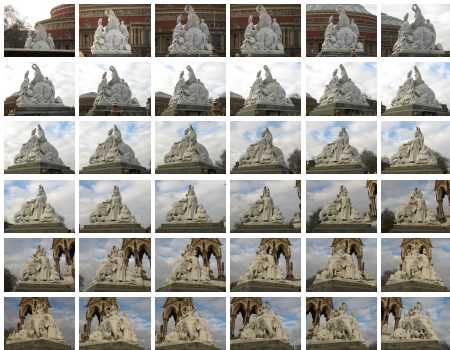
Part I

Course Overview

This Course: Structure from Motion

images + some knowledge about cameras \rightarrow

cameras in 3D + 3D points



36 of 237 images of a memorial



video

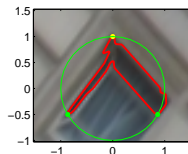
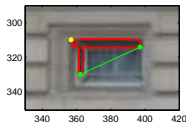
all camera poses, closest 2m, farthest 40m away

Typical phases of a processing pipeline:

1. finding sparse image correspondences
2. recovering camera poses
3. finding dense correspondences
4. surface reconstruction

Phase 1: Sparse Image Correspondences

image features, their descriptors, matches and correspondences



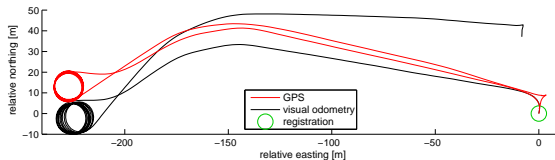
- matches \sim visually similar
- correspondences \sim visually similar and geometrically consistent (yellow)
- finding correspondences must cope with ambiguity
- 5 correspondences determine the relative orientation and translation direction between the ('calibrated') cameras

Phase 2: Recovering Camera Poses

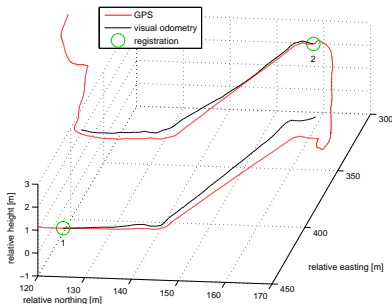
- reversing camera on a car, 30fps; error against RT 3000 GPS system (red)

no fusion with GPS!

Scene I



Scene II



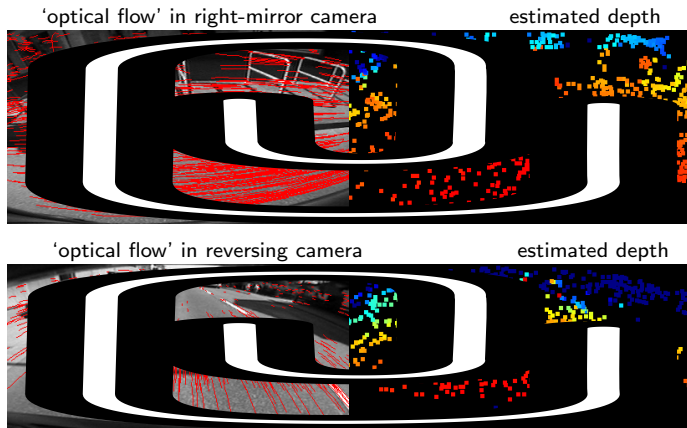
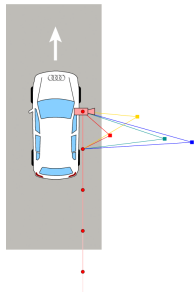
- 1 km, 5% accumulated drift
- measures elevation
- bad lighting conditions
- bad scene

Some applications:

- visual odometry
- SLAM
- the drift is reduced if the correspondences linking camera pairs form a dense graph, not a chain like here

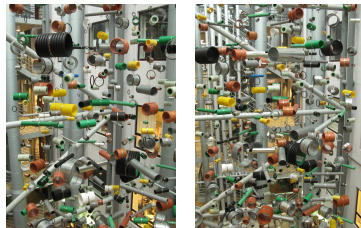
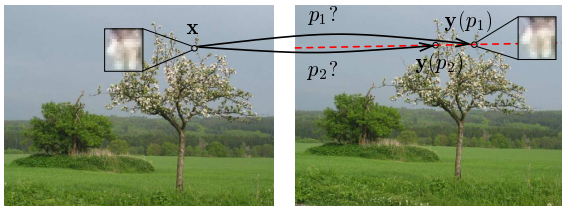
Phase 3a: Semi-Dense Correspondences as a Motion Field

- sensing depth from a single moving camera, 30 fps data stream
standard automotive wide-angle sensor – reversing camera
- moving videocamera \sim time constraint on image match evolution \sim 'optical flow'

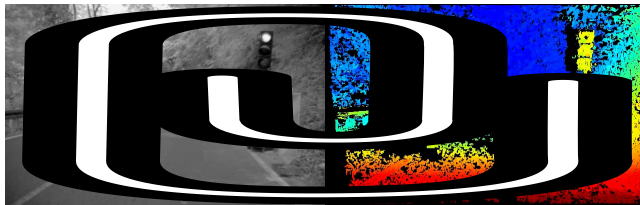
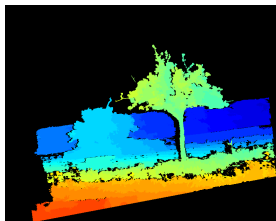


- problems with moving objects (wrong depth)

Phase 3b: Denser Correspondences by Stereovision



Malmö Högskola, Centrum för teknikstudier



video

- the result is a dense 3D point cloud

typically $10^6 - 10^9$ 3D points

Phase 4: Surface Reconstruction

cameras + images



triangulated surface



video



video

Programme:

1. how to do things right in 3D vision [cookbook of effective methods, pitfall avoidance](#)
2. things useful beyond CV [task formulation exercise, powerful robust optimization methods](#)

Content:

- Elements of projective geometry
- Perspective camera
- Geometric problems in 3D vision
- Epipolar geometry and motion-induced homography
- Optimization methods in 3D vision
- 3D structure and camera motion from (many) images
- Stereoscopic vision
- Shape from reflectance

► Important Material Covered Elsewhere

- Homography as a multiview model [H&Z Secs: 2.5, 2.3, 2.4, 4.1, 4.2, A6.1, A6.2]
- Sparse image matching using RANSAC [H&Z Sec. 4.7]

Necessary Prior Knowledge

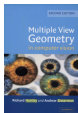
- basic geometry
- elementary linear algebra
vectors, matrices, bases, linear systems of equations, eigensystem, decompositions incl. SVD
- elementary optimization in continuous domain
quadratic problems, constrained optimization, gradient descend, Newton method
- the basics of Bayesian thinking prior, posterior, likelihood, evidence

► Reading

Annotated Slides – this is the reference material not static, evolving over the years

- for deeper digs into geometry, see the GVG lecture notes at <https://cw.felk.cvut.cz/doku.php/courses/a4m33gvv/start>
- there is a Czech-English and English-Czech dictionary for this course

The Book, it will be referenced as [H&Z]



Hartley, R. and Zisserman, A. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd ed, 2003. Secs. 2, 4, 6, 7, 9, 10, 11, 12.

- you can borrow this book from the CMP library
- contact Ms. Radka Kopecka, room G102, <mailto:kopeccka@cmp.felk.cvut.cz>
- indicate you are a student of this course

The Stereo Paper, referenced as [SP]










Šára, R. *Stereoscopic Vision*. 2010

Czech: <http://cmp.felk.cvut.cz/~sara/Teaching/TDV/SP-cz.pdf>

English: <http://cmp.felk.cvut.cz/~sara/Teaching/TDV/SP-en.pdf>

Optional Reading

(available from Google Scholar or CTU Library)

-  M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
-  C. Harris and M. Stephens. A combined corner and edge detector. In *Proc ALVEY Vision Conf*, pp. 147–151, University of Manchester, England, 1988.
-  D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
-  H. Li and R. Hartley. Five-point motion estimation made easy. In *Proc ICPR*, pp. 630–633, 2006.
-  B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. A comprehensive survey of bundle adjustment in computer vision. In *Proc Vision Algorithms: Theory and Practice*, LNCS 1883:298–372. Springer Verlag, 1999.
-  25 years of RANSAC. In *CVPR '06 Workshop and Tutorial [on-line]*, <http://cmp.felk.cvut.cz/ransac-cvpr2006/>. 2006.
-  G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, USA, 3rd edition, 1996.

1. OpenCV (Open Source Computer Vision): A library of programming functions for real time computer vision. [on-line] <http://opencv.willowgarage.com/wiki/>
2. T. Pajdla, Z. Kukelova, M. Bujnak. Minimal problems in computer vision. [on-line] <http://cmp.felk.cvut.cz/minimal/> Last update Oct 22, 2009.
3. Rob Hess. SIFT Feature Detector. [on-line] <http://web.engr.oregonstate.edu/~hess/>. Last update 05/06/2010.
4. Marco Zuliani. RANSAC Toolbox for Matlab. [on-line] <http://vision.ece.ucsb.edu/~zuliani/Code/Code.html>. Last update Oct 18, 2009
5. Manolis Lourakis. A Generic Sparse Bundle Adjustment C/C++ Package based on the Levenberg-Marquardt Algorithm. <http://www.ics.forth.gr/~lourakis/sba/>. Last update Jan 5, 2010.

► Notes on Slide Style

I am using a consistent slide style:

- main material is in black (like this)
- remarks and notes are in small blue font typically flushed to the right like this
- papers or books are referenced like this [H&Z, p. 100] or [Golub & van Loan 1996]
except H&Z or SP, references are to the list on Slide 12
- most references are linked (clickable) in the PDF, the triple of icons ↻↻↻ on the bottom helps you navigate back and forth, they are: back, find, forward
check the references above
- each major part of the lecture starts with a slide listing the equivalent written material
- slides containing examined material have a **bold title** with a marker ► like this slide
- mandatory homework exercises are in small red font, after a circled asterisk
 - ⊛ H1; 10pt: syntax: <problem ID in submission system>; <points>: explanation
deadline: Lecture Day + 2 weeks; unit penalty per 7 days; see the Submission System
- non-mandatory homework exercises are in green
 - ⊛ P1; 1pt: same syntax; deadline: end of semester

Thank You





SECOND EDITION

Multiple View Geometry

in computer vision



Richard Hartley and Andrew Zisserman

CAMBRIDGE

SECOND EDITION

Multiple View Geometry

in computer vision



Richard Hartley and Andrew Zisserman

CAMBRIDGE

How To Teach Stereoscopic Matching?

(Invited Paper)

Radim Šára

Center for Machine Perception

Department of Cybernetics

Czech Technical University in Prague, Czech Republic

Email: sara@cmp.felk.cvut.cz

Abstract—This paper describes a simple but non-trivial semi-ense stereoscopic matching algorithm that could be taught in Computer Vision courses. The description is meant to be instructive and accessible to the student. The level of detail is sufficient for a student to understand all aspects of the algorithm design and to make his/her own modifications. The paper includes the core parts of the algorithm in C code. We make the point of explaining the algorithm so that all steps are derived from first principles in a clear and lucid way. A simple method is described that helps encourage the student to benchmark his/her own improvements of the algorithm.

I. INTRODUCTION

Teaching stereovision in computer vision classes is a difficult task. There are very many algorithms described in the literature that address various aspects of the problem. A good up-to-date comprehensive review is missing. It is thus very difficult for a student to acquire a balanced overview of the area in a short time. For a non-expert researcher who wants to design and experiment with his/her own algorithm, reading the literature is often a frustrating exercise.

This paper tries to introduce a well selected, simple, and reasonably efficient algorithm. My selection is based on a long-term experience with various matching algorithms and on an experience with teaching a 3D computer vision class. I believe that by studying this algorithm the student is gradually educated about the way researchers think about stereoscopic matching. I tried to address all elementary components of the algorithm design, from occlusion modeling to matching task formulation and benchmarking.

An algorithm suitable for teaching stereo should meet a number of requirements:

Simplicity: The algorithm must be non-trivial but easy to implement.

Accessibility: The algorithm should be described in a complete and accessible form.

Performance: Performance must be reasonably good for the basic implementation to be useful in practice.

Education: The student should exercise formal problem formulation. Every part of the algorithm design must be well justified and derivable from first principles.

Development potential: The algorithm must possess potential for encouraging the student to do further development.

Learnability: There should be a simple recommended method for choosing the algorithm parameters and/or selecting their most suitable values for a given application.

Besides the basic knowledge in computer science and algorithms, basics of probability theory and statistics, and introductory-level computer vision or image processing, it is assumed that the reader is familiar with the concept of epipolar geometry and plane homography. The limited extent of this paper required shortening some explanations.

The purpose of this paper is not to give a balanced state-of-the-art overview. The presentation, however, does give references to relevant literature during the exposition of the problem.

II. THE BASIC STRUCTURE OF STEREOSCOPIC MATCHING

Stereoscopic matching is a method for computing disparity maps from a set of images. In this paper we will only consider pairs of cameras that are not co-located. We will often call them the left and the right camera, respectively. A disparity map codes the shift in location of the corresponding local image features induced by camera displacement. The direction of the shift is given by *epipolar geometry* of the two cameras [1] and the magnitude of the shift, called *disparity*, is approximately inversely proportional to the camera-object distance. Some disparity map examples are shown in Fig. 10. In these maps, disparity is coded by color, close objects are reddish, distant objects bluish, and pixels without disparity are black. Such color-coding makes various kinds of errors clearly visible.

This section formalizes object occlusion so that we could derive useful necessary constraints on disparity map correctness. We will first work with spatial points. The 3D space in front of the two cameras is spanned by two pencils of optical rays, one system per camera. The spatial points can be thought of as the intersections of the optical rays. They will be called *possible correspondences*. The task of matching is to accept some of the possible correspondences and reject the others. Fig. 1 shows a part of a surface in the scene (black line segment) and a point p on the surface. Suppose p is visible in both cameras (by optical ray r_1 in the left camera and by optical ray t_1 in the right camera, both cameras are located above the 'scene'). Then every point on ray r_1 that lies in front of p must be transparent and each point of r_1 behind p must be occluded. A similar observation holds for ray t_1 . This shows that the decision on point acceptance and rejection are not independent: If the correspondence given by point p is accepted, then a set of correspondences $X(p)$

How To Teach Stereoscopic Matching?

(Invited Paper)

Radim Šára

Center for Machine Perception

Department of Cybernetics

Czech Technical University in Prague, Czech Republic

Email: sara@cmp.felk.cvut.cz

Abstract—This paper describes a simple but non-trivial semi-ense stereoscopic matching algorithm that could be taught in Computer Vision courses. The description is meant to be instructive and accessible to the student. The level of detail is sufficient for a student to understand all aspects of the algorithm design and to make his/her own modifications. The paper includes the core parts of the algorithm in C code. We make the point of explaining the algorithm so that all steps are derived from first principles in a clear and lucid way. A simple method is described that helps encourage the student to benchmark his/her own improvements of the algorithm.

I. INTRODUCTION

Teaching stereovision in computer vision classes is a difficult task. There are very many algorithms described in the literature that address various aspects of the problem. A good up-to-date comprehensive review is missing. It is thus very difficult for a student to acquire a balanced overview of the area in a short time. For a non-expert researcher who wants to design and experiment with his/her own algorithm, reading the literature is often a frustrating exercise.

This paper tries to introduce a well selected, simple, and reasonably efficient algorithm. My selection is based on a long-term experience with various matching algorithms and on an experience with teaching a 3D computer vision class. I believe that by studying this algorithm the student is gradually educated about the way researchers think about stereoscopic matching. I tried to address all elementary components of the algorithm design, from occlusion modeling to matching task formulation and benchmarking.

An algorithm suitable for teaching stereo should meet a number of requirements:

Simplicity: The algorithm must be non-trivial but easy to implement.

Accessibility: The algorithm should be described in a complete and accessible form.

Performance: Performance must be reasonably good for the basic implementation to be useful in practice.

Education: The student should exercise formal problem formulation. Every part of the algorithm design must be well justified and derivable from first principles.

Development potential: The algorithm must possess potential for encouraging the student to do further development.

Learnability: There should be a simple recommended method for choosing the algorithm parameters and/or selecting their most suitable values for a given application.

Besides the basic knowledge in computer science and algorithms, basics of probability theory and statistics, and introductory-level computer vision or image processing, it is assumed that the reader is familiar with the concept of epipolar geometry and plane homography. The limited extent of this paper required shortening some explanations.

The purpose of this paper is not to give a balanced state-of-the-art overview. The presentation, however, does give references to relevant literature during the exposition of the problem.

II. THE BASIC STRUCTURE OF STEREOSCOPIC MATCHING

Stereoscopic matching is a method for computing disparity maps from a set of images. In this paper we will only consider pairs of cameras that are not co-located. We will often call them the left and the right camera, respectively. A disparity map codes the shift in location of the corresponding local image features induced by camera displacement. The direction of the shift is given by *epipolar geometry* of the two cameras [1] and the magnitude of the shift, called *disparity*, is approximately inversely proportional to the camera-object distance. Some disparity map examples are shown in Fig. 10. In these maps, disparity is coded by color, close objects are reddish, distant objects bluish, and pixels without disparity are black. Such color-coding makes various kinds of errors clearly visible.

This section formalizes object occlusion so that we could derive useful necessary constraints on disparity map correctness. We will first work with spatial points. The 3D space in front of the two cameras is spanned by two pencils of optical rays, one system per camera. The spatial points can be thought of as the intersections of the optical rays. They will be called *possible correspondences*. The task of matching is to accept some of the possible correspondences and reject the others. Fig. 1 shows a part of a surface in the scene (black line segment) and a point p on the surface. Suppose p is visible in both cameras (by optical ray r_1 in the left camera and by optical ray t_1 in the right camera, both cameras are located above the 'scene'). Then every point on ray r_1 that lies in front of p must be transparent and each point of r_1 behind p must be occluded. A similar observation holds for ray t_1 . This shows that the decision on point acceptance and rejection are not independent: If the correspondence given by point p is accepted, then a set of correspondences $X(p)$