Insperata accident magis saepe quam quae speres.
(Things you do not expect happen more often than things you do expect) Plautus (ca 200(B.C.)

Project no: 027787

DIRAC

Detection and Identification of Rare Audio-visual Cues

Integrated Project
IST - Priority 2

DELIVERABLE NO: D1.1
Omnidirectional Sensors and Features for Tracking and 3D Reconstruction

Date of deliverable:31.12.2006
Actual submission date: 26.1.2007

Start date of project: 01.01.2006 Duration: 60 months

Organization name of lead contractor for this deliverable: **Czech Technical University in Prague (CTU)**

Revision [1]

Project co-funded by the European Commission within the Sixth Framework Program (2002-2006)		
Dissemination Level		
PU	Public	
PP	Restricted to other program participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	X
CO	Confidential, only for members of the consortium (including the Commission Services)	

Detection and Identification of Rare Audiovisual Cues

Insperata accident magis saepe quam quae speres.
(Things you do not expect happen more often than things you do expect) Plautus (ca 200(B.C.)

D1.1 Omnidirectional Sensors and Features for Tracking and 3D Reconstruction

Czech Technical University in Prague (CTU)

Abstract:
This deliverable presents work in design and calibration of omnidirectional sensors developed for the DIRAC project for visual data acquisition. Several sensor designs are presented with both off- and on-line calibration methods. On-line calibration method is studied also as a minimal problem using only 8 correspondences for simultaneous estimation of epipolar geometry and radial distortion. Feature detection in omnidirectional images is tackled and a global and local approach to image processing is presented for omnidirectional images. Summary of experimental datasets captured during the first year of DIRAC is given at the end of this deliverable.

Table of Content

1 Introduction

This deliverable summarizes work on omnidirectional sensor design and calibration and features for tracking and 3D reconstruction from omnidirectional images. We advocate omnidirectional sensors as visual sensors for the DIRAC demonstrator since they provide images covering at least hemi-spherical field of view. Such a large field of view is crucial for a system which should react on environment and detect rare events and cannot be provided by conventional cameras. Omnidirectional cameras can be both dioptric (fish eye lenses) and catadioptric (employing mirrors). We use fish eye lenses for the DIRAC project, therefore the words omnidirectional and fish eye will be synonyms in this text.

The aim of this text is to provide only basic technical details on the selected topics. Technical reports with more thorough descriptions of selected problems are attached to this deliverable (Bakstein et al. 2006, Kukelova and Pajdla 2006, Torii and Pajdla 2006). The structure of this deliverable is as follows. Section 2 presents a brief overview of omnidirectional sensors developed for visual data acquisition for the DIRAC project. Calibration methods of these sensors are discussed in Section 3. Minimal problem formulation of estimation of epipolar geometry simultaneously with the radial distortion is presented in Section 4. Global and local approaches to feature extraction from omnidirectional images are described in Section 5. Finally, Section 6 contains a list of all datasets captured during the first year of the DIRAC project.

2 Omnidirectional Sensor Design and their Calibration

We have tested several omnidirectional sensors with the aim to design a prototype for the first DIRAC demonstrator. We have used these sensors to capture datasets presented in Experimental Data Sets 6. Detailed description of the sensors can be found in the attached technical report (Bakstein et al. 2006). Here we present a summary in Table 1. It can be seen that there is a trade-off between the resolution and the frame rate. It has to be decided, whether a lower resolution is suitable for the scene classification task. Also the sensor should be as small as possible while providing images of good quality. Currently, there are two main setups in use, see **Figure 1**. The first one is a pair of Kyocera cameras with a high resolution and very good image quality, but at a slow frame rate and at the cost of a heavy and bulky setup. The second one is a small lightweight setup of two Fire-I cameras, which gives small resolution images at a high frame rate. In the future, we will focus on testing of a pair of Marlin cameras which give a good compromise on image quality and frame rate in a small and light setup.

Table 1 A summary of omnidirectional sensors.

Camera	Lens	FOV	Resolution	Frames/s.
Kyocera M410R	Nikon FC-E9	183°	4Mpix	3
Pixelink	Door peep hole	170°	1.5Mpix	5
Pixelink	Sunex DSL125	185°	0.6Mpix	5
Fire-I	Sunex DSL125	130°(hor.)	0.3Mpix	30
AVT Marlin	Sunex DSL125	185°	1Mpix	7.5

Figure 1: A synchronized stereo pair of two Kyocera cameras (left). A compact stereo pair of two Fire-I cameras (right).

3 Calibration of Omnidirectional Sensors for Partners

We have implemented two different omnidirectional camera calibration techniques. Again, we refer the reader to the attached technical report (Bakstein et al. 2006) for details, here we briefly discuss the methods and their contribution for the DIRAC project.

The first one is an offline calibration (Bakstein 2006) from a known calibration target. This is used to determine the model for an unknown lens or mirror, where we make use of the known geometry of the target, see**Figure 2**. For the partners this means that they have to send the lens or mirror to CTU to calibrate it using our 3D calibration object. Once the model is determined, any addition calibrations (for example when mounting the lens on a different camera) can be performed using the autocalibration approach.

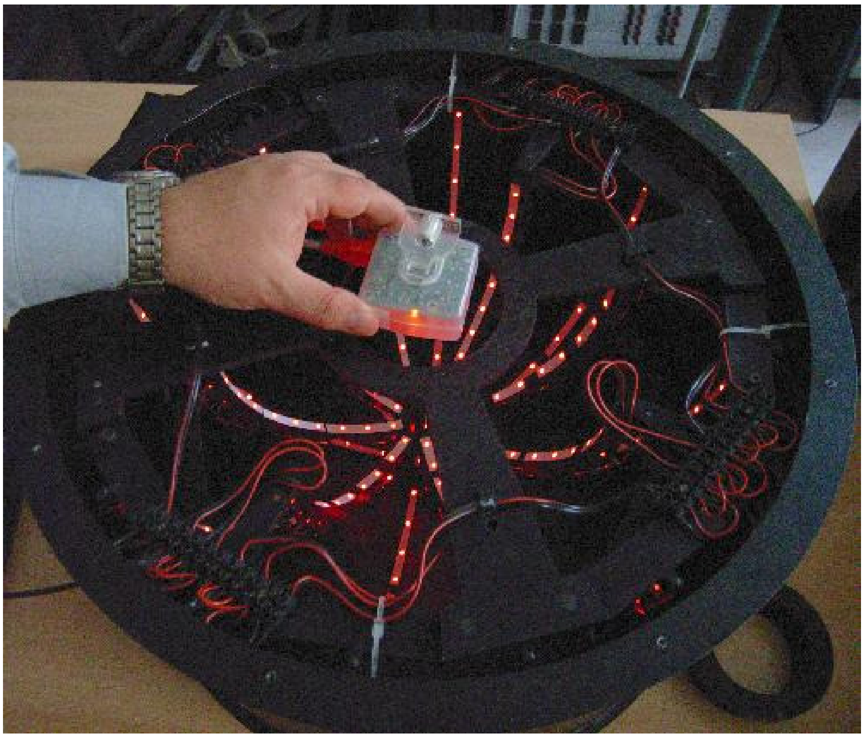


Figure 2: Offline calibration using a known calibration target.

In case the projection model is known, either provided by the manufacturer or from offline calibration, the model parameters can be estimated in an autocalibration procedure (Micusik and Pajdla 2006). This estimation is based on features detected in the images and does not require any calibration target. The whole procedure starts with feature detection in images from the sequence, see Figure 3. Then, epipolar geometry is estimated together with the model parameters from tentative correspondences of the detected features in a RANSAC loop to get consistent matches. The epipolar geometry is computed from the consistent matches and encodes camera position for consecutive image pairs in the sequence and the whole camera trajectory and pose can be recovered, see Figure 4. Therefore, the partners have to send only the sequence to CTU or run our software themselves. CTU also cooperates with KUL on implementation of this autocalibration in the KUL 3D reconstruction module, as it is described in Deliverable 3.1.

Figure 3: Features matched using MSER and APTS (Mikolajczyk and Schmid 2002) features in a fisheye stereo image pair.

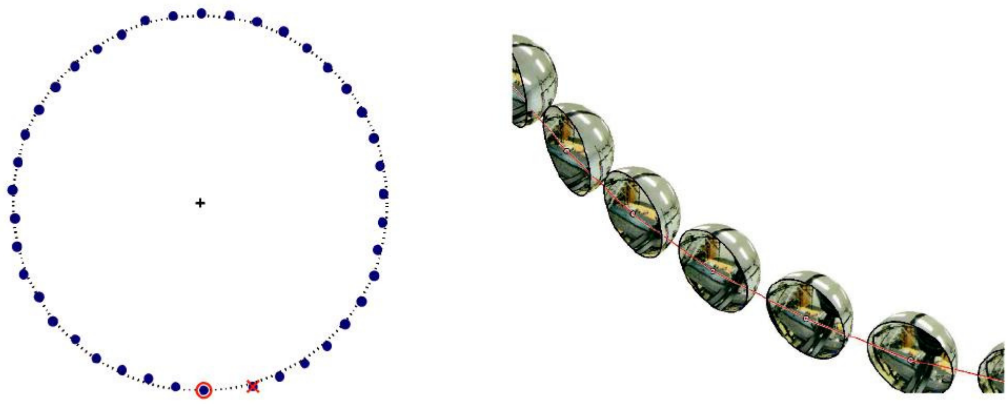


Figure 4: Autocalibration from an image sequence with an inwards looking omnidirectional camera rotating off-center on a circular path. The recovered camera trajectory on the left and estimated camera poses on the right.

4 Minimal Problem Formulation for Radial Distortion Autocalibration

The autocalibration method presented above used nine correspondences for epipolar geometry estimation in the RANSAC loop. However, this is not the smallest number of correspondences for this task, since the method did not make use of the fact, that the fundamental matrix does not have a full rank and therefore the minimal problem for epipolar geometry estimation with radial distortion is to use eight correspondences and the condition that determinant of the fundamental is zero. Smaller number of correspondences significantly reduces the number of samples to be drawn in the RANSAC loop which brings a considerable speed up of the autocalibration procedure.

Unfortunately, the zero determinant condition for the fundamental matrix with radial distortion leads to a non-trivial system of algebraic equations. Our approach, described in detail in (Kukelova and Pajdla 2006), follows the techniques using a Gröbner basis to solve the system of algebraic equations, but instead of computing a full Gröbner basis we compute only the action matrix. From its eigenvectors, the fundamental matrix is computed. To compute the action matrix, the original algebraic equations arising from the zero determinant condition and from the epipolar constraint are transformed into a linear set of equations by treating each monomial as an unknown in the new linear set of equations. From this undetermined set of equations, we get a basis of its null space expressed as a set of polynomial equations, which is simpler than the original set and we compose the action matrix by algebraic manipulation of generators of this set.

The algebraic manipulations described above are performed off-line in advance due to the fact, that in the problem of estimation of fundamental matrices, the combination of monomials is the same regardless on the image measurements, for non-degenerate configurations. As a result, the action matrix is composed offline and concrete values are filled in the estimation process, the rest of the computation is efficient and can be inserted into a RANSAC loop. **Figure 5** illustrates output of the method from (Kukelova and Pajdla 2006) on real data.

Figure 5: Left: fish eye image with significant radial distortion. Right: corrected image. Images from (Kukelova and Pajdla 2006).

5 Features for Omnidirectional Images

We tested different feature detectors directly on omnidirectional images, namely MSER (Perdoch 2004) and APTS (Mikolajczyk and Schmid 2002). It turns out that some scene parts, such as buildings, are better detected by MSER detectors and other parts, such as trees, by the APTS detectors, see **Figure 6**.

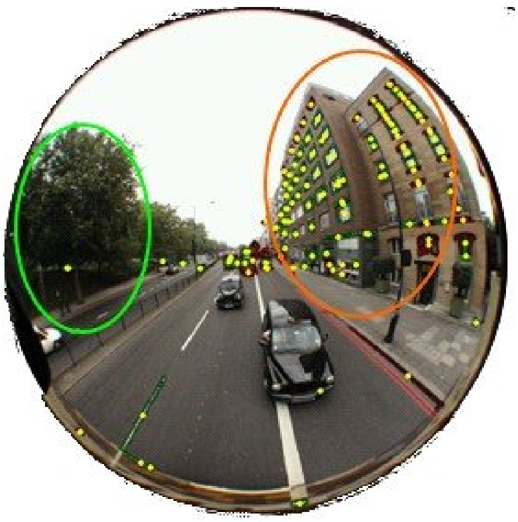


Figure 6: Different feature features should be used for different depending on the character the scene.

Since the APTS take significantly more time to compute and the features are almost complementary, we should be able to select the feature detector based on the scene contents and to run all possible feature detectors on the whole image. This will be one of the topics of our work in cooperation with KUL in the next year of the project.

Omnidirectional images exhibit significant distortion which varies over the field of view, see the top row of circles in Figure 7. As the viewing direction moves towards the edge of the field of view, the circles become more distorted. This makes the feature matching task difficult. For example, MSER features are created from regions, to which an ellipse is fitted, transformed into circle and then a polar sample of this circular region is used to create the region descriptor. Similarity or affine transforms can be use dto map the ellipe into the circle, both do not compensate for the distortion of the omnidirectional image.Some regions have an elliptical shape themselves, but the distortion of omnidirectional regions makes even circular regions strongly elliptical and therefore their descriptors for a position in the centre and at the edge of the field of view differ.

There are two different approaches to tackle this problem. The first one uses global image remapping so that the new image follows a model which does not exhibit distortion dependent on viewing direction. This is illustrated in the bottom row of Figure 7, where the original image was remapped to follow a stereographic projection which is a conformal mapping so that the circles remain circles over the complete field of view. The other approach is a local remapping of image patches to a perspective image with a limited field of view; note that the whole omnidirectional image cannot be mapped to a perspective image.

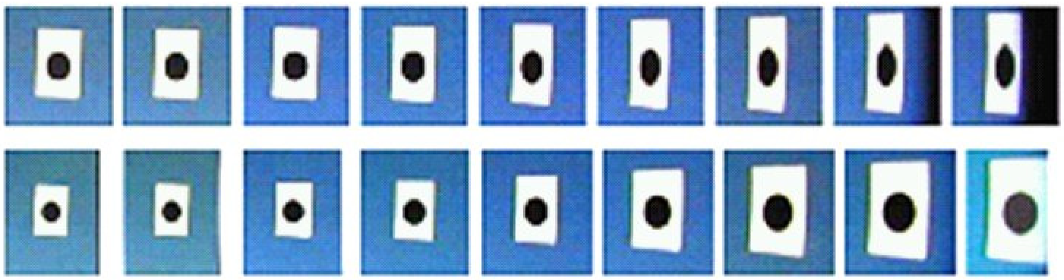


Figure 7: Top row: As a circular target moves over the field of view in an image acquired by a fish eye lens, the circle get distorted. Bottom row: In an image remapped using a stereographic mapping, the circular shape is preserved.

5.1 Global Image Rectification

Global image remapping first lifts the image points to a sphere and then reprojects them to the image plane by a stereographic projection, see Figure 8. For the lifting to a sphere, the model of the fish eye projection has to be known. There are several models for fish eye lenses, see the attached technical report (Bakstein et al. 2006) for details. We used an equidistant projection in our experiments. More details on the lifting of points to the sphere can be found in (Torii and Pajdla 2006).

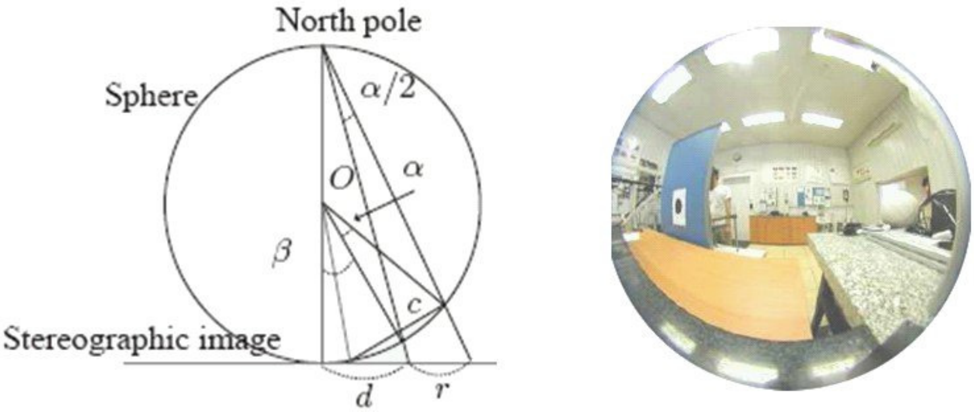


Figure 8: Left: fish eye image is first mapped to a sphere and then stereographic projection is used to create a conformal image. Right: An image from the experiment illustrated in Figure 7. Image from (Torii and Pajdla 2006).

The effect of the proposed method on feature descriptions can be seen in **Figure 9**. The top row shows images at the centre and edge of the field of view for both original and stereographic image. For matching, a circular region around the features is converted into a polar patch using similarity resp. affine transform; the polar patches are shown in the bottom row. It can be seen that the polar patch after the similarity transform contains also pixels from the surrounding of the region and that number of these pixels is smaller for the stereographic image.

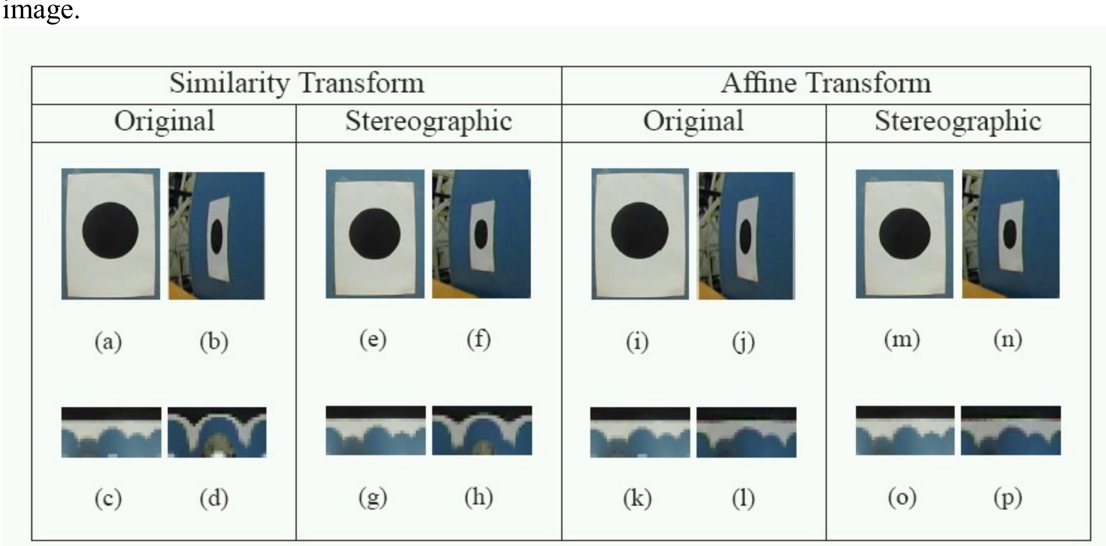


Figure 9: Selected distinguished region for a fish eye image (Original) and the fish eye image transformed a stereographic image captured at the center (a,e,i,m) and at the edge (b,f,j,n) of the field of view. Similarity (left) and affine (right) transform was used to relate the region descriptors. Stereographic image gives a small visual improvement over the original image for both transforms. Image from (Torii and Pajdla 2006).

The improvement in similarity of the patches for the affine transform is clearer from comparison of the values of affine coefficients l_1 and l_2 of all descriptors shown in **Figure 10** and **Figure 11** respectively. The figures show affine coefficients for all detected distinguished regions (used to compute the MSERs) and for tentative regions, which are selected as match candidates. The ratio of the affine coefficients of the tentative regions is closer to 1 for the stereographic image, which gives better results in matching.

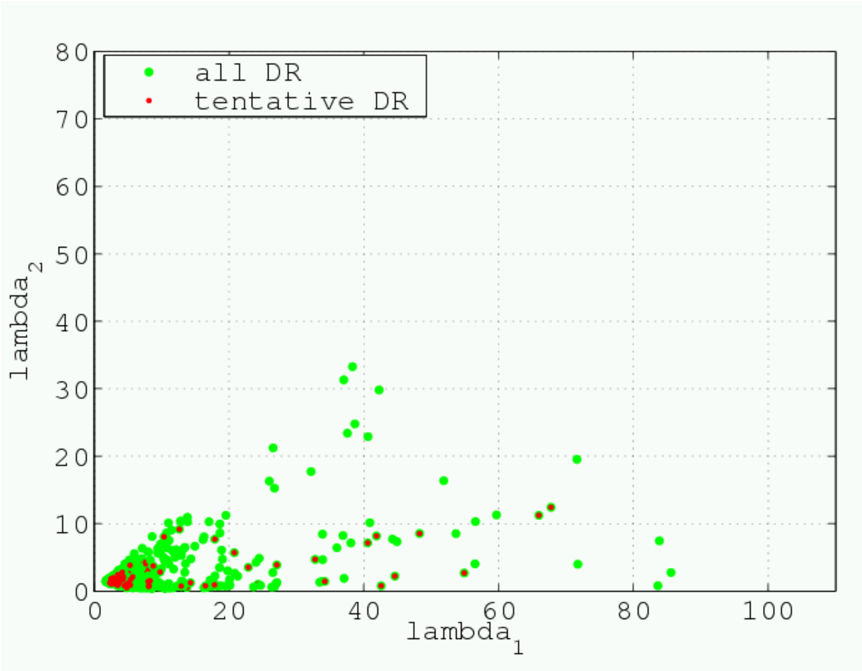


Figure 10: l_1 and l_2 of all distinguished regions and tentative correspondences detected in a fish eye image.

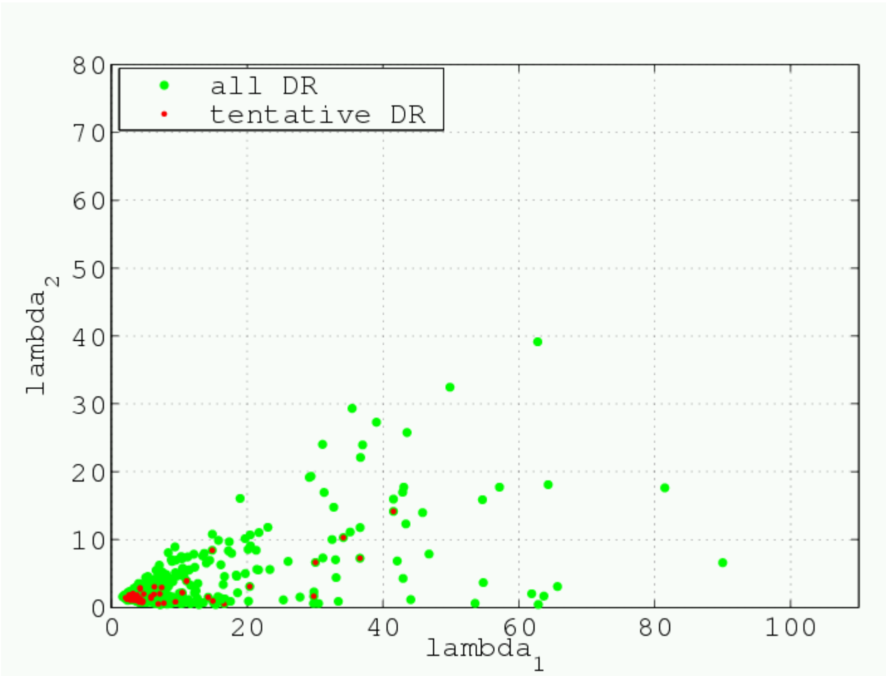


Figure 11: l_1 and l_2 of all distinguished regions and tentative correspondences detected in a stereographic image.

5.2 Local Perspective Mapping

5.2.1 *Perspective Patches for Scene Classification*

For classification of the type of object in the scene, image patches are compared with learned samples in a database. For omnidirectional images, the patches are obtained as small perspective cut-outs from the omnidirectional image. The centres of these cut-outs were obtained by a sphere tessellation with icosahedron as a starting surface as midpoints of the triangles approximating the sphere. The number of subdivisions of triangular sides on the icosahedron determines the number of patches and should be in correspondence with the image resolution and patch size. For example, in the case of DIRAC -CMPdata-05 dataset, the input images have resolution of 750 ´562 pixels and the optimal patch size for subdivision level 4 is 13 pixels, resulting in 624 patches. We used 16 x 16 pixel patches for convenience.

The next level in the Gaussian pyramid is created by filtering the input image by a 13 ´13 Gaussian filter with s=1.5 and scaling down to 369 ´276 image. The scaling factor is a ratio of angles between centres (respective their representations as using vectors on a sphere) of the two neighbouring patches for tessellation of the sphere with subdivision level 4 (for the input image) and 3, for the filtered image.

The last level in the Gaussian pyramid is created with the same filtering and scaling computed again as a ratio of angles between the centres of two neighbouring patches. The resulting down scaled image has size 174 ´ 130 pixels and the subdivision level for the tessellation is 2. All patches extracted from the pyramid are shown in **Figure 12**.

Figure 12: Image patches for different levels of the image pyramids.

It should be noted that fisheye images can be mapped to a sphere, not to a plane. Therefore, the patches are computed using spherical coordinates, with azimuth corresponding to an angle in the fisheye image and elevation maximal p/2 in the centre of this image, zero at the edges. The patches do not have the same orientation; they are transformed so that azimuth corresponds to the rows and elevation to the columns. This means that the objects in the patches are rotated, depending on their position in the field of view.

5.2.2 Perspective Patches for Feature Matching

The same idea, to generate local perspective patches for some viewing direction, can be used for feature matching in epipolar geometry estimation. As it was mentioned before at the beginning of this section, omnidirectional images are generally not conformal and object shapes vary significantly over the field of view. In feature matching, the algorithms assume that the features are related by affine transform, which does not adequately model the variance in shapes in omnidirectional images. Our approach to this problem was to modify our wide-baseline matching code so that the feature descriptions are generated from small perspective cut-outs instead of from patches from the original omnidirectional image.

The correspondence between different the levels in the pyramid is as follows. At the lowest level, the first patch corresponds to the first 4 patches at the higher level. First from these 4 patches corresponds to first 4 patches at the top level. The second one to the next 4 and so on. This follows from the tessellation, where each triangle of the initial shape, the icosahedron, is subdivided into four triangles. This subdivision is then repeated recursively.

6 Experimental Data Sets

Table 2 summarizes all datasets acquired during the first year of the project. We begun with tests of different acquisition hardware, produced data for experiments with various features for scene classification, and last but not least, we captured first dataset with video and audio, in cooperation with the Oldenburg group. All experimental data sets are published on our DIRAC project page: http://cmp.felk.cvut.cz/projects/dirac/.

Table 2 Summary of datasets acquired by CTU during the first year of the DIRAC project.

DIRAC -CMPData-01	Walking Omni-Camera, Kyocera Camera + Nikon Lens, video and audio, not synchronized
DIRAC -CMPData-02	Walking Omni-Camera, Kyocera Camera + Nikon Lens, every 10-th frame from the sequence
DIRAC -CMPData-03	Walking Omni-Camera, Kyocera Camera + Nikon Lens, Images (jpg) and regions saved in Matlab matrices.
DIRAC -CMPData-04	Walking Stereo Omni-Camera, 2 x Kyocera Camera + Nikon Lens, Images (jpg) and regions saved in Matlab matrices for both cameras.
DIRAC -CMPData-05	Image patches from images in the DIRAC -CMPdata-03 dataset.
DIRAC -CMPData-06	Image from the test of the Marlin camera in CMP.
DIRAC -CMPData-07	First dataset with video and audio, acquired in Leuven during the DIRAC workshop.

6.1 First data set with audio and video data

We captured one data set which contains both audio and video. Audio stream was recorded by the Oldenburg team using dummy hearing aids. Video data was recorded using a pair of Fire-I cameras set to 15 frames/sec. A timestamp was assigned to each image, which allows to detect dropped frames. A crude synchronization of audio and video streams by hand clapping was used in the experiment. We plan to detect visual features in the images while IDIAP team will detect audio features in the short pieces of audio data centred at the time of the respective image, determined from the timestamps. Thus, for a known time, there will be a combined audio and visual descriptor which we plan to use for rare even detection in the future work.

7 Conclusion

We have designed and tested various sensors for the DIRAC project and captured several datasets for our partners. We have also done research in the field of feature detection in omnidirectional images, with both global and local approaches. Calibration of the omnidirectional cameras became a natural part of our work since it is a key to handling images from such cameras. We may conclude that the goals for the first year were reached.

8 Reference

Hynek Bakstein**.** "Non-central Cameras for 3D Reconstruction". *PhD Thesis CTU–CMP–2006–09, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republ.,* 2006.

Hynek Bakstein, Michal Havlena, Petr Pohl, and Tomáš Pajdla. "Omnidirectional sensors and their calibration for the DIRAC project". *Research Report CTU–CMP–2006–13, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, December,* 2006. **(See annexes)**

Zuzana Kúkelová and Tomáš Pajdla. "Solving polynomial equations for minimal problems in computer vision". *Research Report CTU–CMP–2006–10, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, December,* 2006. **(See annexes)**

Branislav Mičušík and Tomáš Pajdla. "Structure from motion with wide circular field of view cameras". *IEEE Trans. Pattern Anal. Mach. Intell., 28(7):1135–1149,* 2006.

K. Mikolajczyk and C. Schmid. "An affine invariant interest point detector". *In Proceedings of the European Conference of Computer Vision (ECCV), volume 1, pages 128–142*. 2002.

Michal Perďoch. "Invariant detection of stable regions in 2D images". *Master's thesis, Czech Technical University, Prague, Czech Republic.* 2004.

Akihiko Torii and Tomáš Pajdla. "Image matching insensitive to the change of camera orientation". *Research Report CTU–CMP–2006–11, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, December*. 2006. **(See annexes)**

9 Annexes

See next pages for articles

CENTER FOR
MACHINE PERCEPTION

CZECH TECHNICAL
UNIVERSITY

RESEARCH REPORT

ISSN 1213-2365

Omnidirectional sensors and their calibration for the Dirac project

(Version 1.0)

Hynek Bakstein
Michal Havlena
Petr Pohl
Tomáš Pajdla

bakstein@cmp.felk.cvut.cz

CTU–CMP–2006–13

Available at
ftp://cmp.felk.cvut.cz/pub/cmp/articles/bakstein/CMP-TR-2006-13.pdf

This work is supported by the European IST Programme Project FP6-0027787. This paper only reflects the authors' views and funding agencies are not liable for any use that may be made of the information contained herein.

Research Reports of CMP, Czech Technical University in Prague, No. 13, 2006

Published by

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Omnidirectional sensors and their calibration for the Dirac project

Hynek Bakstein Michal Havlena Petr Pohl
Tomáš Pajdla

1 Introduction

This report describes various omnidirectional sensor designs developed for the Dirac project. The aim is to create a light weight omnidirectional setup with a stero pair of cameras. Calibration procedure for omnidirectional cameras is presented both as offline calibration from a known 3D target and autocalibration from image sequences. The report is structured as follows. Section 2 discusses the sensor design and parameters. Section 3 presents models of omnidirectional sensors and calibration approaches, from simple approximation to complex model.

2 Omnidirectional sensors

Omnidirectional sensors capture a wide field of in a single image and provide peripheral view. Peripheral view is essential for a system which should react to changing environment. In the following, we present omnidirectional sensors designed for the Dirac project.

2.1 Omni-stereo rig

We designed an omnidirectional stereo camera rig consisting of two consumer digital cameras — Kyocera M410R equipped with a Nikon FC-E9 fisheye adapter. These cameras can capture 4Mpix images at a frame rate of 3 frames per second and the fisheye adapter provides 183° field of view. Our colleague Pavel Krsek has built a remote control box, shown in Figure 1 bottom, which allows to release both shutters at the same time and also to lock exposures of both cameras at once or separately for each cameras. The remote control is connected to the cameras by wires soldered directly to the camera shutters. A schematic digram of the remote control is shown in Figure 2.

Figure 1: Omnidirectional stereo rig.

2.2 Compact omnidirectional camera

Our aim is to design a compact light weight system with an omnidirectional optics and a 1Mpix resolution camera with a frame rate at least 15 frames/second in case of a single camera and 7.5 frames/second for a stereo pair. At first, we presented a design employing a door peep hole and a Pixelink camera, shown in Figure 3(a). Door peep hole is compact enough but does not have good optical properties.

Next, we have selected an appropriate lens, the Sunex DSL125 fisheye lens with a 180° field of view [3]. This lens is suitable for a megapixel camera with a 1/2" sensor size and provides good image quality. Its main drawback is that it has to be mounted very close to the imager, 4.31 mm. Since the lens is very compact, a C-mount adapter has to be used and because of the small distance between the exit pupil of the lens and the images, the C-mount thread on the camera has to be shortened.

With a compact lens, a compact camera should be used too. We have two options, one is a cheap Fire-I camera [1], shown in Figure 3(b), which can be directly fitted with the Sunex lens. It has a good frame rate but low resolution and image quality. The other option is an AVT Marlin camera [7], with a good resolution and image quality, which has a C-mount thread and has to be adapted to accommodate the Sunex lens. We will discuss both cameras later in this report.

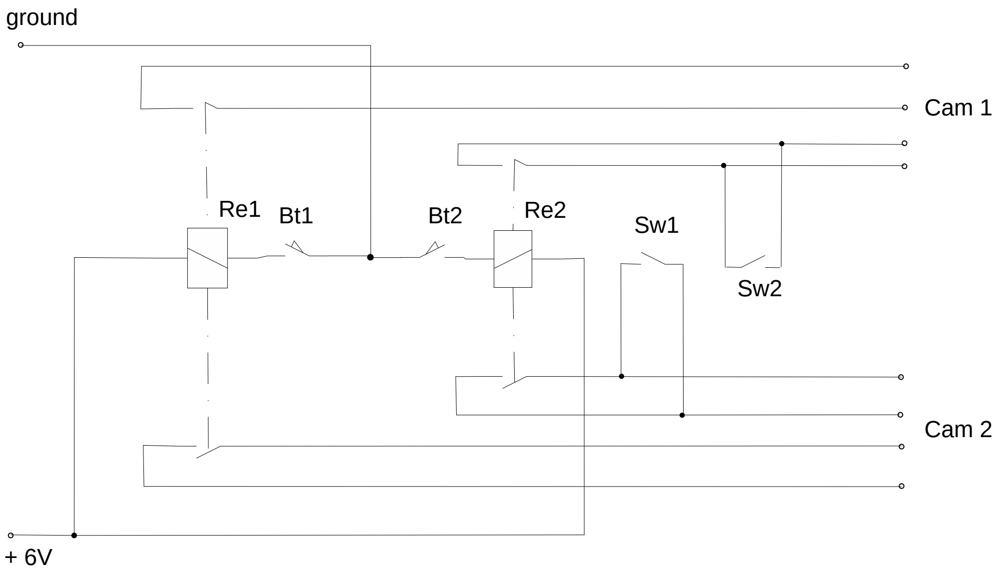


Figure 2: Diagram of the remote control for two Kyocera cameras.

2.3 Synchronization of the camera pair

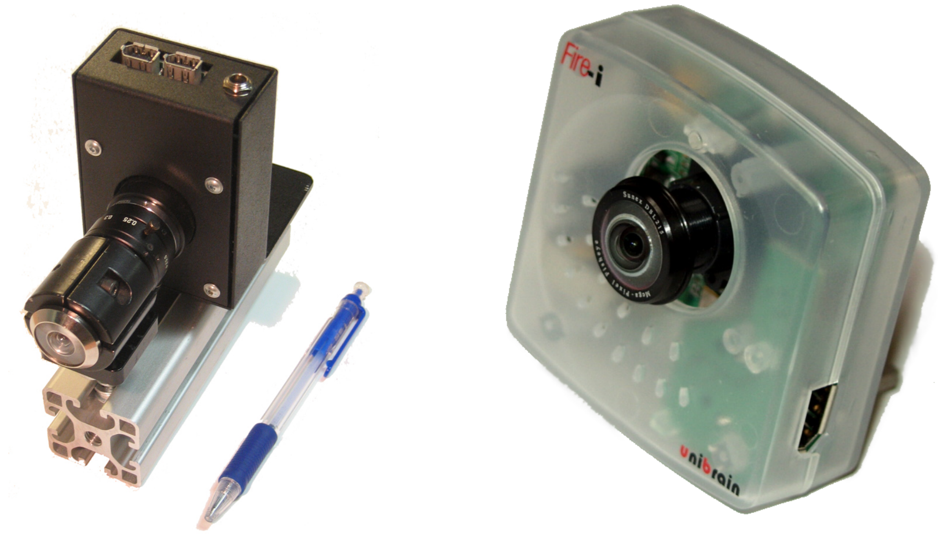
An important issue is the synchronization of the two cameras in the stereo pair. For the two Kyocera cameras, there is a synchronized shutter release implemented in the remote control but this assures only that the two cameras start recording at the same time. For the rest of the sequence we have to rely on a constant frame rate. We made experiments with a counter and found this synchronization satisfactory. Eventual dropped frames can be detected from the trajectory estimation run for each camera separately, since the frame rate is slow and the cameras move significantly between the frames.

Different situation is with the Fire-I cameras which do not have any hardware shutter control. We have to rely on their high frame rate and on the time stamps assigned to each frame by our capturing application once it is captured. The camera are capable of 30 frames per second, which means that the worst case difference between two frames is 33 milliseconds. This should be sufficient for our purpose.

A solution to the synchronization problem is to use an external trigger for the cameras. The Marlin camera, which we plan to use in the future experiments, supports external triggering.

2.4 Summary of the sensor designs

Table 1 summarizes the camera/lens setups we designed and tested.



(a) (b)

Figure 3: (a) A small omnidirectional camera from a door peep hole. (b) Sunex DSL125 fisheye lens on a Fire-i camera.

Camera	Lens	FOV	Resolution	Frames/s.
Kyocera M410R	Nikon FC-E9	183°	4Mpix	3
Pixelink	Door peep hole	170°	1.5Mpix	5
Pixelink	Sunex DSL125	185°	0.6Mpix	5
Fire-I	Sunex DSL125	130° (hor.)	0.3Mpix	30
AVT Marlin	Sunex DSL125	185°	1Mpix	7.5

Table 1: A summary of omnidirectional sensors.

2.5 Sensor specifications for the first Dirac demonstrator

We have tested the following system and we propose it as the first design for the Dirac demonstrator:

- Fire-i Color Board Camera (with RAW),
- Sunex DSL 125, and
- Firewire card - Kouwell L1582V (no power connector) or Kouwell 7002V (with power connector and USB 2.0).

A stereo pair of the Fire-I cameras is capable of capturing images at a frame rate up to 15 frames/second (each) and is depicted in Figure 4.

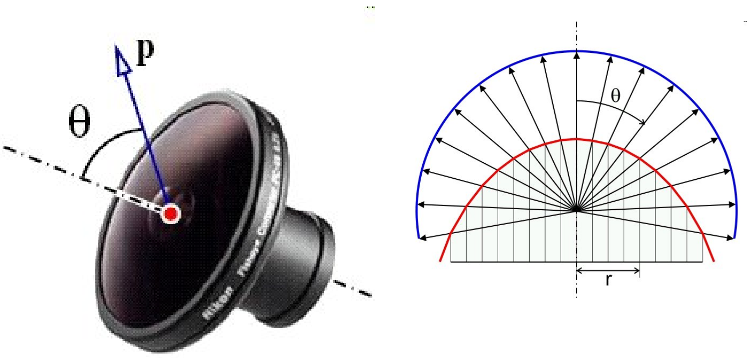
We have also made preliminary tests with AVT Marlin F-146C camera provided by KUL. We succeeded in connecting the Marlin camera to our

Figure 4: A stereo rig of two Fire-I cameras with the Sunex DSL125 fisheye lenses.

image acquisition software and to capture testing dataset. What has to be done is a synchronization and modification of the lens mount to accommodate the Sunex DSL125 lens, the tests were performed with Nikon FC-E8 (predecessor to FC-E9 used on the Kyocera cameras) fish eye adapter. However, the Marlin camera provides images with higher resolution and better quality than the Fire-I camera and we plan to use it in the final design of the Dirac demonstrator.

3 Omnidirectional camera calibration

Fish eye images exhibit significant radial distortion. Moreover, when the field of view approaches to 180°, perspective projection cannot be used to model the fish eye image since it involves tangent of the angle between the light ray and the optical axis, as it is depicted in Figure 5(a). From the figure also follows that we assume the projection function to be radially symmetric, that is to map light rays with a constant angle θ into circle with a radius r, as illustrated in Figure 5(b).



(a) (b)

Figure 5: (a) Fisheye projection is parameterized by angles θ between light rays p and the optical axis. (b) Projection models then map the angle θ to image radius r.

3.1 Approximation by equidistant projection

Figure 6: RANSAC-based circle fitting used for estimation of the maximal radius of the fisheye image.

The first approach to handling a fish eye image is to use a simple equidistant projection model, which can be written as

$$r = a\theta \ , \tag{1}$$

where a is the only model parameter. This model maps constant increments in the angle θ to constant increments of image radius r. This model can be easily initialized if the field of view of the lens θ_{max} and the maximal radius in the image r_{max} is known. The first value is usually provided by the manufacturer of the lens and the second has to be measured in the image. We usually work with circular fish eye lenses where a circle corresponding to θ_{max} is visible. Due to its uneven boundary, vignetting effects, and parasitic light, some robust circle fitting approach has to be used to get r_max. We use a RANSAC based method on a thresholded image, see Figure 6.

3.2 Model determination using an off-line calibration target

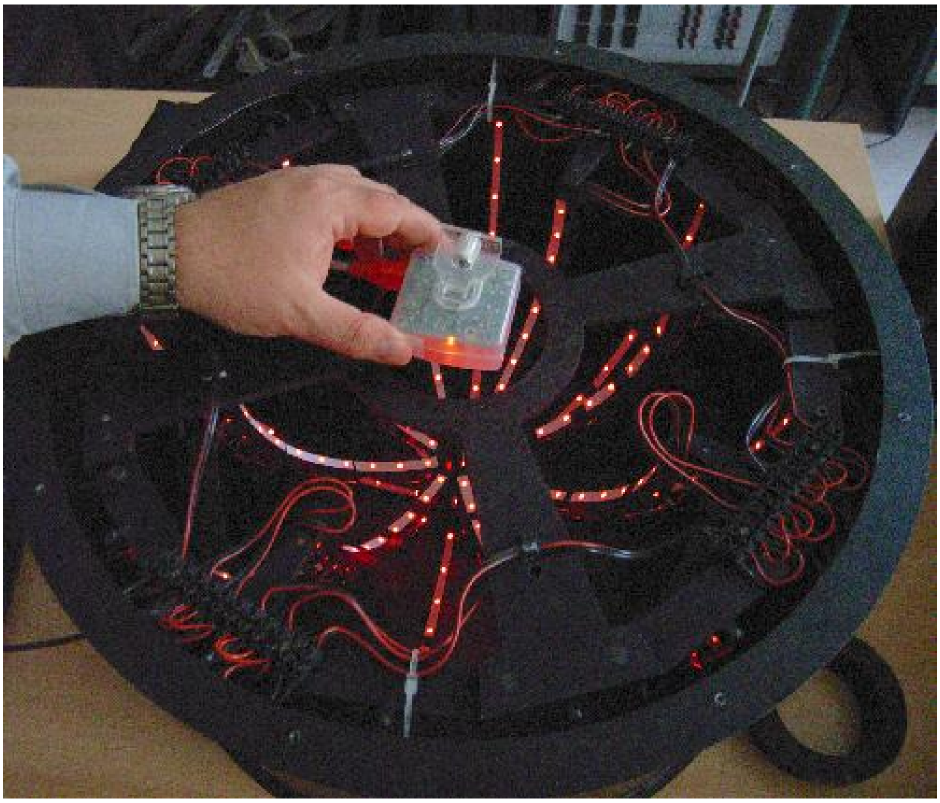


Figure 7: Offline calibration using a known calibration target.

Different models are used for the fish eye lens designs to suit a particular task. The sensor calibration therefore begins with selection of a model which approximates the best the lens. It can be either provided by the manufacturer or estimated using our calibration target, shown in Figure 7. The choice of a model does not only depend on the lens, but also the task and precision required. For a RANSAC based outlier rejection a simple one parametric model, such as equidistant projection, described above, is sufficient.

When higher precision is needed, we can use a model with more parameters, such as equi-solid angle projection

$$r = a \sin \frac{\theta}{b}$$

or stereographic projection

$$r = a \tan \frac{\theta}{b}$$

or a polynomial approximation of the unknown true function [2]. These models have a big disadvantage that they do not have a direct inverse function, a lookup table has to be used. Therefore, we use the following model for autocalibration, described in depth in [5] and briefly in the next section. A projection of a scene point into the image is given by the following equation.

$$r = \frac{a - \sqrt{a^2 - 4\theta^2 b}}{2b\theta} \tag{2}$$

The projection equation has an inverse, the reprojection equation, describing the relation between the image point and the light ray.

$$\theta = \frac{ar}{1 + br^2} \tag{3}$$

We have implemented Matlab functions of the above projection and reprojection equations for our partners, available at our Dirac pages. The functions allow to measure angle between light rays defined by two image points and to create a perspective cut-out from a part of the fish eye image, as it is depicted in Figure 8.

3.3 Autocalibration from images

The division model has another advantage, besides the direct inverse function. It can be included into a formulation of the epipolar constraint so that the fundamental matrix describes not only the mutual position of two cameras but also the radial distortion.

Epipolar geometry defines a relation between two corresponding image points x and x' as the familiar epipolar constraint, where $\mathbf{F}$ is the fundamental matrix.

$$x\mathbf{F}x' = 0$$

Moreover, each point defines a so called epipolar line, which is a projection of the light rays respective to the point into the second image. The above

(a) (b)

Figure 8: (a) An image acquired with a Sunex DSL125 lens. (b) A perspective cut-out from the fisheye image (a).

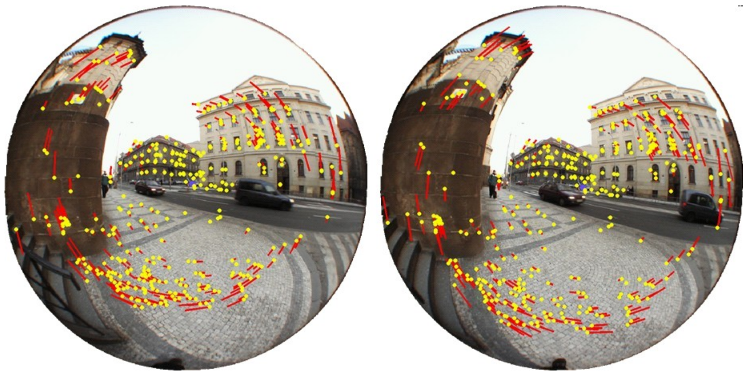


Figure 9: Matched points in a pair of fisheye images satisfying the epipolar constraint. Yellow lines are magnified distances from the epipolar curves.

formulation was written for perspective cameras but holds also for calibrated fish eye lenses (**F** then becomes the essential matrix **E**), but the epipolar lines become curves [6]. However, we can still evalute distance between the points and the epipolar curves, as it is dpicted in Figure 9.

A concept of fundamental matrix exists also for fish eye lenses, where it accommodates also the distortion parameters. In [6], it is shown that the epipolar constraint leads to a Polynomial Eigenvalue Problem (PEP),

$$(D_1 + aD_2 + d^2 D_3 + a^3 D_4 + d^4 D_5)\mathbf{f} = 0 \ ,$$

where D_i is composed from image coordinates of detected points, a is a model parameter and $\mathbf{f}$ contains elements of the fundamental matrix $\mathbf{F}$. PEP can be efficiently solved, in Matlab it is implemented in a function `polyeig`.

The equation 2 is a formulation of a two-parametric model. In the same manner as mentioned above with the equidistant projection, a simpler one-parametric formulation of the division model can be used for fast outlier rejection in the RANSAC loop. The advantage is that for the one-parametric model 9 correspondences are required for a direct estimation using the PEP formulation while the two-parametric model has to be estimated using 15 correspondences [6]. Moreover, the two-parametric model cannot be directly transformed into a PEP formulation, it has to be at first linearized [6]. This linearization is based on partial derivatives of the model function at appropriate values of the model parameters. Equivalently to the approximation by the equidistant projection, the maximal image radius r_{max} is used to initialize the first parameter and the second is put equal to zero [6].

We have implemented both algorithms, for the 9-point and for the 15-point formulation. To further speed up the RANSAC estimation, even smaller number of correspondences can be used, since the 9-point algorithm ignores the fact that the fundamental matrix is singular and a zero determinant condition can be used to reduce the number of correspondences to eight, which represent a minimal problem formulation for simultaneous estimation of epipolar geometry and radial distortion [4].

3.4 Model refinement

The linearization of the two-parametric model uses some values at which partial derivatives of the model function are computed. This choice affects the precision of the model estimation. The best results were obtained by local optimization of the two-parametric model using an output from offline calibration as the initial estimate. Results from the off-line optimization on using the calibration target are also not precise due to imprecision of the target assembly, but their values provide a good start for the autocalibration. Figure 10 shows the difference between models estimated off-line and after refinement. Visual illustration of this difference is depicted in Figures 11 and 12 which show a fish eye image mapped onto a cube, with the sides of the cube being perspective images.

4 Conclusion

We have summarized omnidirectional sensors developed for the Dirac project. W focused on creating a light weight setup with a pair of cameras equipped with a fish eye lens and achieving a frame rate of 15 frames/second. We developed two functional prototypes, the first one with a high resolution and the second one meeting the requirements, but with a low resolution. Our future work will focus on a final setup for the first Dirac demonstrator which will use the Marlin cameras, with sufficient resolution.

We also discussed the calibration procedures for fish eye lenses, starting with a simple approximation by an equidistant projection, ending with an

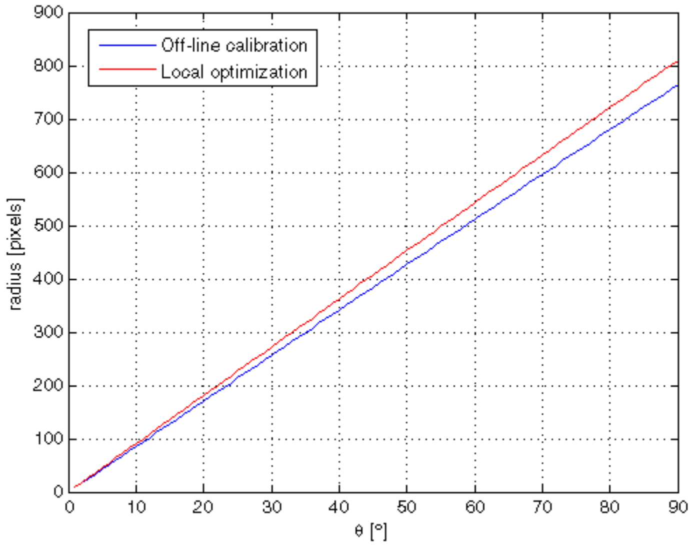


Figure 10: Difference between model parameters estimated in the offline calibration procedure and locally optimized parameters from a real sequence.

autocalibration procedure which uses off-line computed parameters as an initial value. Function for manipulation with fish eye images have been implemented and made available for the partners in the project.

References

[1] Unibrain S. A. `http://www.unibrain.com/Products/VisionImg/Fire_i_BC.htm`.

[2] Hynek Bakstein. *Non-central Cameras for 3D Reconstruction*. PhD Thesis CTU–CMP–2006–09, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, March 2006.

[3] Sunex Inc. `http://www.sunex.com/`.

[4] Zuzana Kúkelová and Tomáš Pajdla. Solving polynomial equations for minimal problems in computer vision. Research Report CTU–CMP–2006–10, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, December 2006.

[5] Branislav Mičušík. *Two-View Geometry of Omnidirectional Cameras*. Phd thesis, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University, Prague, Czech Republic, June 2004.

Figure 11: Fisheye image mapped on a cube using parameters obtained by offline calibration.

Figure 12: Fisheye image mapped on a cube using parameters obtained by local optimization.

[6] Branislav Micusik and Tomas Pajdla. Structure from motion with wide circular field of view cameras. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(7):1135–1149, 2006.

[7] Allied Vision Technologies. `http://www.alliedvisiontec.com/produktinfos.html?t=produktinfos&o=51&a=selectid`.

CENTER FOR MACHINE PERCEPTION

CZECH TECHNICAL UNIVERSITY

RESEARCH REPORT

ISSN 1213-2365

Solving polynomial equations for minimal problems in computer vision

Zuzana Kúkelová
Tomáš Pajdla

kukelova@cmp.felk.cvut.cz pajdla@cmp.felk.cvut.cz

CTU–CMP–2006–10

January 17, 2007

The author was supported by the EC project FP6-IST-027787 DIRAC. Any opinions expressed in this paper do not necessarily reflect the views of the European Community. The Community is not liable for any use that may be made of the information contained herein.

Research Reports of CMP, Czech Technical University in Prague, No. 10, 2006

Published by

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Solving polynomial equations for minimal problems in computer vision

Zuzana Kúkelová Tomáš Pajdla

January 17, 2007

Abstract

Epipolar geometry and relative camera pose computation are examples of tasks which can be formulated as minimal problems, i.e. they can be solved from a minimal number of image points. Finding the solution leads to solving systems of algebraic equations. Often, these systems are not trivial and therefore special algorithms have to be designed to achieve numerical robustness and computational efficiency. In this work we suggest improvements of current techniques for solving systems of polynomial equations suitable for some vision problems. We introduce two tricks. The first trick helps to reduce the number of variables and degrees of the equations. The second trick can be used to replace computationally complex construction of Gröbner basis by a simpler procedure. We demonstrate benefits of our technique by providing a solution to the problem of estimating radial distortion and epipolar geometry from eight correspondences in two images. Unlike previous algorithms, which were able to solve the problem from nine correspondences only, we enforce the determinant of the fundamental matrix be zero. This leads to a system of eight quadratic and one cubic equation. We provide an efficient and robust solver of this problem. The quality of the solver is demonstrated on synthetic and real data.

1 Introduction

Estimating camera models from image matches in an important problem. It is one of the oldest computer vision problems and even though much has already been solved some questions remain still open. For instance, a number of techniques for modeling and estimating projection models of wide angle lenses [6, 18, 10, 28, 29] appeared recently. Often in this case, the projection is modeled as the perspective projection followed by radial "distortion" in the image plane.

Figure 1: (Left) Pair of images with radial distortion. (Right) Corrected images.

Methods for estimating distortion parameters may be divided into three groups (1) the plumbline methods, (2) the 3D-2D point correspondences methods and (3) the 2D-2D point correspondences methods.

The plumbline methods estimate radial distortion parameters using only one input image. They are based on the fact that the straight lines in the world are distorted into curves in the image. Therefore, they require a presence of straight lines in the scene which must be identified in the image [1, 4, 13, 27].

The second group of methods estimates radial distortion parameters using correspondences between points in the scene with known 3D coordinates and their 2D projections in the image [32, 30]. They again need only one image, however their disadvantage is that they need to use a known 3D calibration object.

The third group of methods are methods which use image point correspondences. These method need to have two or more views of the scene. Their big advantage is that no scene structure has to be known a priori. Some of these methods form the problem of estimating radial distortion parameters as a minimization problem [21, 31] and use some iterative nonlinear optimization algorithms, e.g.

Levenberg-Marquardt, to solve it. These methods have classical problems of iterative nonlinear optimization algorithms.

The particularly interesting formulation, based on the division model [6], has been introduced by Fitzgibbon. His formulation leads to solving a system of algebraic equations. It is especially nice because the algebraic constraints of the epipolar geometry, $\det(\mathtt{F}) = 0$ for an uncalibrated and $2\,\mathtt{E}\,\mathtt{E}^\top\mathtt{E} - \mathrm{trace}(\mathtt{E}\,\mathtt{E}^\top)\mathtt{E} = 0$ for a calibrated situation [11], can be "naturally" added to the constraints arising from correspondences to reduce the number of points needed for estimating the distortion and the fundamental matrix. A smaller number of points considerably reduces the number of samples in RANSAC [5, 11]. This is the good news. The bad news is that the resulting systems of polynomial equations are more difficult than, e.g., the systems arising from similar problems for estimating epipolar geometry of perspective cameras [24, 23].

In this work we will solve the problem arising from taking $\det(\mathtt{F}) = 0$ constraint into account.

Fitzgibbon [6] did not use the algebraic constraints on the fundamental matrix. In fact, he did not explicitly pose his problem as finding a solution to a system of algebraic equations. Thanks to neglecting the constraints, he worked with a very special system of algebraic equations which can be solved numerically by using a quadratic eigenvalue solver. In this way he simultaneously estimate a multiple view geometry and radial distortion from nine point correspondences.

Micusik and Pajdla [18] also neglected the constraints when formulating the estimation of paracatadioptric camera model from image matches as a quartic eigenvalue problem. The work [20] extended Fitzgibbon's method for any number of views and any number of point correspondences using generalized quadratic eigenvalue problem for rectangular matrices, again without explicitly solving algebraic equations.

Li and Hartley [15] treated the original Fitzgibbon's problem as a system of algebraic equations and used the hidden variable technique [2] to solve them. This technique allows to use different algebraic distortion models with different number of parameters. Authors of this work mentioned the possibility of using the singularity of the fundamental matrix as a algebraic constraint which allows to estimate two distortion parameters from minimum nine point correspondences but they didn't practically use this constraint. Using this constraint makes the problem much harder because the degree of equations involved significantly increases. Therefore, for estimating more then one parameter they collect groups of nine point correspondences to obtain sufficient number of equations just from epipolar constraints. This leads to problems of solving systems of more polynomial equations of higher degree. Thus, this algorithm is suitable for computing only few parameters like one-parameter division model. The resulting technique [15] for one-parameter division model solves exactly the same problem as [6] but in a different way. Our experiments have shown that the quality of the result was comparable but the technique [15] was considerably slower than the original technique [6].

The division model has also been used by [29]. They have proposed a linear approach that can estimate any number of radial distortion parameters of the division model by estimating the radial trifocal tensor using seven correspondences across three views. However, this method works only for a camera observing a plane or undergoing a pure rotation.

Claus and Fitzgibbon [7] introduced a new rational function model for radial lens distortion in wide-angle and catadioptric lenses and have proposed a method for estimating distortion parameters of this model from point correspondences.

In our work we formulate the problem of estimating the radial distortion from image matches as a system of algebraic equations and by using the constraint $\det(\mathtt{F}) = 0$ we get a minimal solution to the autocalibration of radial distortion from eight correspondences in two views.

Our work adds a new minimal problem solution to the family of previously developed minimal problems, e.g. the perspective three point problem [5, 9], the five point relative pose problem [19, 23, 16], the six point focal length problem [24, 14], and six point generalized camera problem [25].

These problems often lead to a nontrivial systems of algebraic equation. For solving such nontrivial systems there doesn't exist one general robust and computationally efficient method. Therefore in recent years various algorithms based on algebraic geometry concepts have been proposed. Most of these algorithms are based on two important algebraic concepts: Gröbner basis and resultants. There exist many papers and books which deal with algebraic methods for solving systems of polynomial equations and Gröbner basis and resultants concepts including theory and applications, like works of Mourrain, Emiris, Cox i.e. [2, **?**, **?**]

Stewénius [22] proposed a Gröbner basis method to solve some minimal problems in computer vision. Using this method Stewénius et al. solved problems like five point relative pose problem [23], six point focal length problem [24], six point generalized camera problem [25] and other problems. This solver is a variation of the classical Gröbner basis technique [2], which is a standard algebraic technique for solving polynomial systems. Stewenius solver is based on the fact that in a class of problems the way of computing the Gröbner basis is always the same and for particular data these Gröbner bases differ only in coefficients and on the fact that the algorithm for computation of the Gröbner basis can be realized using Gauss-Jordan elimination.

Another technique was used in [16] and [14] where a simple algorithms based on the hidden variable resultant technique to solve two minimal problems, five point relative pose problem and six point focal length problem has been proposed. These problems were previously solved by Stewenius and Nister using Gröbner basis and Gauss-Elimination techniques [23, 24, 19]. Comparing to the Gröbner basis algorithms, the hidden-variable algorithms are easier to understand and to implement.

We follow the general paradigm for solving minimal problems in which a problem is formulated as a set of algebraic equations which need to be solved. Our main contribution is in improving the technique for solving the set of algebraic equations proposed in [22] and applying it to solve the minimal problem for the autocalibration of radial distortion. We use the algebraic constraint $\det(\mathtt{F}) = 0$ on the fundamental matrix to get an 8-point algorithm. It reduces the number of samples in RANSAC and is more stable than previously known 9-point algorithms [6, 15].

2 Solving algebraic equations

In this section we will introduce the technique we use for solving systems of algebraic equations. We use the nomenclature from excellent monographs [3, 2], where all basic concepts from polynomial algebra, algebraic geometry, and solving systems of polynomial equations are explained.

Our goal is to solve a system of algebraic equations $f_1(x) = ... = f_m(x) = 0$ which are given by a set of m polynomials $F = \{f_1, ..., f_m | \, f_i \in \mathbb{C}[x_1, ..., x_n]\}$ in n variables over the field $\mathbb{C}$ of complex numbers. We are only interested in systems which have a finite number, say N, solutions and thus $m \geq n$.

The ideal I generated by polynomials F can be written as

$$I = \left\{ \sum_{i=1}^{m} f_i \, p_i \, | \, p_i \in \mathbb{C}[x_1, ..., x_n] \right\}$$

with $f_1, ..., f_m$ being generators of I. The ideal contains all polynomials which can be generated as an algebraic combination of its generators. Therefore, all polynomials from the ideal are zero on the zero set $Z = \{x | f_1(x) = ... = f_m(x) = 0\}$. In general, an ideal can be generated by many different sets of generators which all share the same solutions. There is a special set of generators though, the reduced Gröbner basis $G = \{g_1, ..., g_l\}$ w.r.t. the lexicographic ordering, which generates the ideal I but is easy (often trivial) to solve. Computing this basis and "reading off" the solutions from it is the standard method for solving systems of polynomial equations. Unfortunately, for most computer vision problems this "Gröbner basis method w.r.t. the lexicographic ordering" is not feasible because it has double exponential computational complexity in general.

To overcome this problem, a Gröbner basis G under another ordering, e.g. the graded reverse lexicographical ordering, which is often easier to compute, is constructed. Then, the properties of the *quotient ring* $A = \mathbb{C}[x_1, ..., x_n]/I$, i.e. the set of equivalence classes represented by remainders modulo I, can be used to get the solutions. The linear basis of this quotient ring can be written as $B = \{\mathbf{x}^\alpha | \mathbf{x}^\alpha \notin \langle LM(I) \rangle\} = \left\{\mathbf{x}^\alpha | \overline{\mathbf{x}^\alpha}^G = \mathbf{x}^\alpha\right\}$, where $\mathbf{x}^\alpha$ is monomial $\mathbf{x}^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} ... x_n^{\alpha_n}$, $\overline{\mathbf{x}^\alpha}^G$ is the reminder of $\mathbf{x}^\alpha$ on the division by G, and $\langle LM(I) \rangle$ is ideal generated by leading monomials of all polynomials form I. In many cases (when I is radical [2]), the dimension of A is equal to the number of solutions N. Then, the basis of A consists of N monomials, say $B = \left\{\mathbf{x}^{\alpha(1)}, ..., \mathbf{x}^{\alpha(N)}\right\}$. Denoting the basis as $b(\mathbf{x}) = \left[\mathbf{x}^{\alpha(1)} ... \mathbf{x}^{\alpha(N)}\right]^T$, every polynomial $q(\mathbf{x}) \in A$ can be expressed as $q(\mathbf{x}) = b(\mathbf{x})^T c$, where c is a coefficient vector. The multiplication by a fixed polynomial $f(\mathbf{x})$ (a polynomial in variables $\mathbf{x} = (x_1, ..., x_n)$) in the quotient ring A then corresponds to a linear operator $T_f : A \to A$ which can be described by a $N \times N$ action matrix M_f. The solutions to the set of equations can be read off directly from the eigenvalues and eigenvectors of the action matrices. We have

$$f(\mathbf{x})\, q(\mathbf{x}) = f(\mathbf{x}) \left(b(\mathbf{x})^T c\right) = \left(f(\mathbf{x})\, b(\mathbf{x})^T\right) c$$

Using properties of the action matrix M_f, we obtain

$$\left(f(\mathbf{x})\, b(\mathbf{x})^T\right) c = b(\mathbf{x})^T M_f c,$$

Each polynomial $t \in \mathbb{C}[x_1, ..., x_n]$ can be written in the form $t = \sum_{i=1}^{l} h_i g_i + r$, where g_i are basis vectors $g_i \in G = \{g_1, ..., g_l\}$, $h_i \in \mathbb{C}[x_1, ..., x_n]$ and r is the reminder of t on the division by G.

If $\mathbf{p} = (p_1, ..., p_n)$ is a solution to our system of equations, then we can write

$$f(\mathbf{p})\, q(\mathbf{p}) = \left(f(\mathbf{p})\, b(\mathbf{p})^T\right) c = \sum_{i=1}^{l} h_i(\mathbf{p})\, g_i(\mathbf{p}) + r(\mathbf{p})$$

where $r(\mathbf{p})$ is the reminder of $f(\mathbf{p})\, q(\mathbf{p}) = \left(f(\mathbf{p})\, b(\mathbf{p})^T\right) c$ on the division by G.

Because $g_i(\mathbf{p}) = 0$ for all $i = 1, ..., l$ we have $\sum_{i=1}^{l} h_i(\mathbf{p})\, g_i(\mathbf{p}) + r(\mathbf{p}) = r(\mathbf{p})$ and therefore

$$\left(f(\mathbf{p})\, b(\mathbf{p})^T\right) c = r(\mathbf{p}) = \overline{\left(f(\mathbf{p})\, b(\mathbf{p})^T\right) c}^G.$$

So for a solution $\mathbf{p}$, we have

$$\overline{\left(f(\mathbf{p})\, b(\mathbf{p})^T\right) c}^G = \left(f(\mathbf{p})\, b(\mathbf{p})^T\right) c = b(\mathbf{p})^T M_f c$$

for all c, and therefore

$$f(\mathbf{p})\, b(\mathbf{p})^T = b(\mathbf{p})^T M_f.$$

Thus, if $\mathbf{p} = (p_1, ..., p_n)$ is a solution to our system of equations and $f(\mathbf{x})$ is chosen such that the values $f(\mathbf{p})$ are distinct for all $\mathbf{p}$, the N left eigenvectors of the action matrix M_f are of the form

$$v = \beta b(\mathbf{p}) = \beta \left[\mathbf{p}^{\alpha(1)} ... \mathbf{p}^{\alpha(N)}\right]^T,$$

for some $\beta \in \mathbb{C}, \beta \neq 0$.

Thus "action" matrix $\mathtt{M}_f$ of the linear operator $T_f \colon A \to A$ of the multiplication by a suitably chosen polynomial f w.r.t. the basis B of A can be constructed and then the solutions to the set of equations can then be read off directly from the eigenvalues and eigenvectors of this action matrix [2].

2.1 Simplifying equations by lifting

The complexity of computing an action matrix depends on the complexity of polynomials. It is better to have the degrees as well as the number of variables low. Often, original generators F may be transformed into new generators with lower degrees and fewer variables. Next we describe a particular transformation method—*lifting method*—which proved to be useful.

Assume m polynomial equations in l monomials. The main idea is to consider each monomial that appears in the system of polynomial equations as an unknown. In this way, the initial system of polynomial equations of arbitrary degree becomes linear in the new "monomial unknowns". Such system can by written in a matrix form as

$$\mathtt{M} X = 0$$

where X is a vector of l monomials and $\mathtt{M}$ is a $m \times l$ coefficient matrix.

If $m < l$, then a basis of $m - l$ dimensional null space of matrix $\mathtt{M}$ can be found and all monomial unknowns can be expressed as linear combinations of basic vectors of the null space. The coefficients of this linear combination of basic vectors become new unknowns of the new system which is formed by utilizing algebraic dependencies between monomials. In this way we obtain a system of polynomial equations in new variables. The new set of variables consists of unknown coefficients of linear combination of basic vectors and of old unknowns which we need for utilizing dependencies between monomials. The new system is equivalent to the original system of polynomial equations but may be simpler. This abstract description will be made more concrete in section 3.1 or from the following simple example:

Consider a system of 4 quadratic equations in 4 unknowns x_1, x_2, x_3, x_4 and in 6 monomials: $x_1x_3, x_2x_4, x_4, x_3, x_2, x_1$. Write the $i-th$ polynomial $c_{i1}x_1x_3 + c_{i2}x_2x_4 + c_{i3}x_4 + c_{i4}x_3 + c_{i5}x_2 + c_{i6}x_1$, where $c_{ij} \in \mathbb{C}$. Write

$$\begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} \\ c_{21} & c_{22} & c_{23} & c_{24} & c_{25} & c_{26} \\ c_{31} & c_{32} & c_{33} & c_{34} & c_{35} & c_{36} \\ c_{41} & c_{42} & c_{43} & c_{44} & c_{45} & c_{46} \end{pmatrix} \begin{pmatrix} x_1x_3 \\ x_2x_4 \\ x_4 \\ x_3 \\ x_2 \\ x_1 \end{pmatrix} = 0$$

Consider each monomial that appears in the system as an unknown. We have a system of four linear equations in six "monomial" unknowns. In general, we can find a 2-dimensional null space of matrix M. Denote the basic vectors of the null space by $\mathbf{b}_1 = (b_{11,}b_{12,}..., b_{16})^T$ and $\mathbf{b}_2 = (b_{21,}b_{22,}..., b_{26})^T$. We can write:

$$\begin{pmatrix} x_1x_3 \\ x_2x_4 \\ x_4 \\ x_3 \\ x_2 \\ x_1 \end{pmatrix} = a_1 \begin{pmatrix} b_{11} \\ b_{12} \\ b_{13} \\ b_{14} \\ b_{15} \\ b_{16} \end{pmatrix} + a_2 \begin{pmatrix} b_{21} \\ b_{22} \\ b_{23} \\ b_{24} \\ b_{25} \\ b_{26} \end{pmatrix}$$

for some unknown coefficients $a_1, a_2 \in \mathbb{C}$ and known $b_{ij} \in \mathbb{C}$, $i = 1, 2$ and $j = 1, ..., 6$. Coefficients a_1, a_2 become new unknowns of the new system which we form by utilizing the dependencies between monomials. We know that $X_1 = x_1x_3$, $X_4 = x_3$, $X_6 = x_1$. Thus $X_1 = X_4X_6$ and $\acute{X}_2 = X_3X_5$. We get

$$\begin{aligned} a_1b_{11} + a_2b_{21} &= (a_1b_{14} + a_2b_{24})(a_1b_{16} + a_2b_{26}) \\ a_2b_{12} + a_2b_{22} &= (a_1b_{13} + a_2b_{23})(a_1b_{15} + a_2b_{25}) \end{aligned}$$

or equivalently

$$\begin{aligned} b_{14}b_{16}a_1^2 + (b_{14}b_{26} + b_{16}b_{24})\, a_1a_2 + b_{24}b_{26}a_2^2 - a_1b_{11} - a_2b_{21} &= 0 \\ b_{13}b_{15}a_1^2 + (b_{13}b_{25} + b_{15}b_{23})\, a_1a_2 + b_{23}b_{25}a_2^2 - a_1b_{12} - a_2b_{22} &= 0 \end{aligned}$$

In this way we obtain two new quadratic equations in two new unknowns a_1 and a_2 which are equivalent to the original four quadratic equations in 4 unknowns x_1, x_2, x_3, x_4 but they are simpler to solve.

2.2 Constructing action matrix efficiently

The standard method for computing action matrices requires to construct a complete Gröbner basis and the linear basis B of the algebra A and to compute $T_f\left(\mathbf{x}^{\alpha(i)}\right) = \overline{f\mathbf{x}^{\alpha(i)}}^G$ for all $\mathbf{x}^{\alpha(i)} \in B = \left\{\mathbf{x}^{\alpha(1)}, ..., \mathbf{x}^{\alpha(N)}\right\}$ [2]. Note that $\mathbf{x}^{\alpha(i)} = x_1^{\alpha_1(i)} x_2^{\alpha_2(i)} ... x_n^{\alpha_n(i)}$. For some problems, however, it may be very expensive to find a complete Gröbner basis. Fortunately, to compute $\mathtt{M}_f$ we do not always need a complete Gröbner basis. Here we propose a method for constructing the action matrix assuming that the monomial basis B of algebra A is known or can be computed for a class of problems in advance.

Many minimal problems in computer vision have the convenient property that the monomials which appear in the set of initial generators F are always same irrespectively from the concrete coefficients arising from non-degenerate image measurements. For instance, when computing the essential matrix from five points, there always need to be five linear linearly independent equations in elements of $\mathtt{E}$ and ten higher order algebraic equations $2\,\mathtt{E}\,\mathtt{E}^\top\mathtt{E} - \text{trace}(\mathtt{E}\,\mathtt{E}^\top)\,\mathtt{E} = 0$ and $\det(\mathtt{E}) = 0$ which do not depend on particular measurements. Therefore, the leading monomials of the corresponding Gröbener basis, and thus the monomials in the basis B are always the same. They can be found once in advance. To do so, we use the approach originally suggested in [24, 22, 23] for computing Gröbner bases but we retrieve the basis B and polynomials required for constructing the action matrix instead.

Having B, the action matrix can be computed as follows. If for some $\mathbf{x}^{\alpha(i)} \in B$ and chosen f, $f\mathbf{x}^{\alpha(i)} \in A$, then $T_f\left(\mathbf{x}^{\alpha(i)}\right) = \overline{f\mathbf{x}^{\alpha(i)}}^G = f\mathbf{x}^{\alpha(i)}$ and we are done. For all other $\mathbf{x}^{\alpha(i)} \in B$ for which $f\mathbf{x}^{\alpha(i)} \notin A$ consider polynomials $q_i = f\mathbf{x}^{\alpha(i)} + h_i$ from I with $h_i \in A$. For these $\mathbf{x}^{\alpha(i)}$, $T_f\left(\mathbf{x}^{\alpha(i)}\right) = \overline{f\mathbf{x}^{\alpha(i)}}^G = \overline{q_i - h_i}^G = -h_i \in A$. Since polynomials q_i are from the ideal I, we can generate them as algebraic combinations of the initial generators F. Write $h_i = \sum_{j=1}^N c_{ji}\mathbf{x}^{\alpha(j)}$ for some $c_{ji} \in \mathbb{C}$, $i = 1, ..., N$. Then the action matrix M_f has the form

$$M_f = \begin{pmatrix} c_{11} & c_{12} & . & . & . & c_{1N} \\ c_{21} & . & & & & . \\ . & & . & & & . \\ . & & & . & & . \\ . & & & & . & . \\ c_{N1} & c_{N2} & & & & c_{NN} \end{pmatrix}.$$

To get this action matrix $\mathtt{M}_f$, it suffice to generate polynomials $q_i = f\mathbf{x}^{\alpha(i)} + \sum_{j=1}^N c_{ji}\mathbf{x}^{\alpha(j)}$ from the initial generators F for all these $\mathbf{x}^{\alpha(i)} \in B$. This in general seems to be as difficult as generating the Gröbner basis but we shall see that it is quite simple for the problem of calibrating radial distortion which we describe in the next section. It is possible to generate q_i's by starting with F and systematically generating new polynomials by multiplying them by individual variables and reducing them by the Gauss-Jordan elimination. This technique is a variation of the F4 algorithm for constructing Gröbner bases [8] and seems to be applicable to more vision problems. We are currently investigating it and will report more results elsewhere.

2.3 The solver

The algorithmic description of our solver of polynomial equations is as follows.

1. Assume a set $F = \{f_1, ..., f_m\}$ of polynomial equations.
2. Use the lifting to simplify the original set of polynomial equations if possible. Otherwise use the original set.
3. Fix a monomial ordering (The graded reverse lexicographical ordering is often good).
4. Use Macaulay 2 [22] to find the basis B as the basis which repeatedly appears for many different choices of random coefficients. Do computations in a suitably chosen finite field to speed them up.
5. For suitably chosen polynomial f construct the polynomials q_i by systematically generating higher order polynomials from generators F. Stop when all q_i's are found. Then construct the action matrix $\mathtt{M}_f$.
6. Solve the equations by finding eigenvectors of the action matrix. If the initial system of equations was transformed, extract the solutions to the original problem.

This method extends the Gröbner basis method proposed in [24, 22] (i) by using lifting to simplify the problem and (ii) by constructing the action matrix without constructing a complete Gröbner basis. This brings an important advantage for some problems. Next we will demonstrate it by showing how to solve the minimal problem for correcting radial distortion from eight point correspondences in two views.

3 A minimal solution for radial distortion

We want to correct radial lens distortion using the minimal number of image point correspondences in two views. We assume one-parameter division distortion model [6]. It is well known that for standard uncalibrated case without considering radial distortion, 7 point correspondences are sufficient and necessary to estimate the epipolar geometry. We have one more parameter, the radial distortion parameter λ. Therefore, we will need 8 point correspondences to estimate λ and the epipolar geometry. To get this "8-point algorithm", we have to use the singularity of the fundamental matrix $\mathtt{F}$. We obtain 9 equations in 10 unknowns by taking equations from the epipolar constraint for 8 point correspondences

$$\mathbf{p}_{u_i}^\top(\lambda)\,\mathtt{F}\,\mathbf{p}'_{u_i}(\lambda) = 0, \quad i = 1, \dots, 8$$

and the singularity of $\mathtt{F}$

$$\det(\mathtt{F}) = 0,$$

where $\mathbf{p}'_u(\lambda), \mathbf{p}_u(\lambda)$ represent homogeneous coordinates of a pair of undistorted image correspondences.

The one-parameter division model is given by the formula

$$\mathbf{p}_u \sim \mathbf{p}_d/(1 + \lambda r_d^2)$$

where λ is the distortion parameter, $\mathbf{p}_u = (x_u, y_u, 1)$, resp. $\mathbf{p}_d = (x_d, y_d, 1)$, are the corresponding undistorted, resp. distorted, image points, and r_d is the radius of $\mathbf{p}_d$ w.r.t. the distortion center. We assume that the distortion center has been found, e.g., by [10]. We also assume square pixels, i.e. $r_d^2 = x_d^2 + y_d^2$. To use the standard notation, we write the division model as

$$\mathbf{x} + \lambda\mathbf{z} = \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix} + \lambda \begin{pmatrix} 0 \\ 0 \\ r_d^2 \end{pmatrix} \sim \begin{pmatrix} x_u \\ y_u \\ 1 \end{pmatrix}.$$

3.1 Reducing 9 to 7 unknowns by lifting

We simplify the original set of equations by lifting. The epipolar constraint gives 8 equations with 15 monomials (nine 1^{st}order, five 2^{nd}order, one 3^{rd} order)

$$(x_i + \lambda z_i)^T\,\mathtt{F}\,(x'_i + \lambda z'_i) = 0,\; i = 1, ..., 8$$

$$x_i^T\mathtt{F}x'_i + \lambda\left(x_i^T\mathtt{F}z'_i + z_i^T\mathtt{F}x'_i\right) + \lambda^2 z_i^T\mathtt{F}z'_i = 0,\; i = 1, ..., 8$$

$$\mathtt{F} = \begin{pmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{pmatrix}$$

We consider each monomial as an unknown and obtain 8 homogenous equations linear in the new 15 monomial unknowns. These equation can be written in a matrix form

$$\mathtt{M}\,X = 0$$

where $X = (f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, \lambda f_3, \lambda f_6, \lambda f_7, \lambda f_8, \lambda f_9, \lambda^2 f_9,)^T$ and M is the coefficient matrix.

If we denote the i-th row of the matrix M as m_i and write

$$x_i + \lambda z_i = \begin{pmatrix} x_d \\ y_d \\ 1 \end{pmatrix} + \lambda \begin{pmatrix} 0 \\ 0 \\ r_d^2 \end{pmatrix},$$

then $m_i = (x_d x'_d,\ x_d y'_d,\ x_d,\ y_d x'_d,\ y_d y'_d,\ y_d,\ x'_d,\ y'_d,\ 1,\ x_d r'^2_d,\ x_{yd} r'^2_d,\ r_d^2 x'_d,\ r_d^2 y'_d,\ r_d^2 + r'^2_d,\ r_d^2 r'^2_d)$.

We obtain 8 linear equations in 15 unknowns. So, in general we can find 7 dimensional null-space. We write

$$X = x_1 N_1 + x_2 N_2 + x_3 N_3 + x_4 N_4 + x_5 N_5 + x_6 N_6 + x_7 N_7$$

where $N_1, ..., N_7 \in \mathbb{R}^{15\times 1}$ are basic vectors of the null space and $x_1, \ldots, x_7$ are coefficients of the linear combination of the basic vectors. Assuming $x_7 \neq 0$, we can set $x_7 = 1$. Then we can write

$$X = \sum_{i=1}^{7} x_i N_i = \sum_{i=1}^{6} x_i N_i + N_7$$

$$X_j = \sum_{i=1}^{6} x_i N_{ij} + N_{7j}, \quad j = 1, .., 15$$

Considering dependencies between monomials and $\det(\mathtt{F}) = 0$ we get 7 equations for 7 unknowns $x_1, x_2, x_3, x_4, x_5, x_6, \lambda$:

$$X_{10} = \lambda.X_3 \Rightarrow \sum_{i=1}^{6} x_i N_{i,10} + N_{7,10} = \lambda \sum_{i=1}^{6} x_i N_{i,3} + N_{7,3}$$

$$X_{11} = \lambda.X_6 \Rightarrow \sum_{i=1}^{6} x_i N_{i,11} + N_{7,11} = \lambda \sum_{i=1}^{6} x_i N_{i,6} + N_{7,6}$$

$$X_{12} = \lambda.X_7 \Rightarrow \sum_{i=1}^{6} x_i N_{i,12} + N_{7,12} = \lambda \sum_{i=1}^{6} x_i N_{i,7} + N_{7,7}$$

$$X_{13} = \lambda.X_8 \Rightarrow \sum_{i=1}^{6} x_i N_{i,13} + N_{7,13} = \lambda \sum_{i=1}^{6} x_i N_{i,8} + N_{7,8}$$

$$X_{14} = \lambda.X_9 \Rightarrow \sum_{i=1}^{6} x_i N_{i,14} + N_{7,14} = \lambda \sum_{i=1}^{6} x_i N_{i,9} + N_{7,9}$$

$$X_{15} = \lambda.X_{14} \Rightarrow \sum_{i=1}^{6} x_i N_{i,15} + N_{7,15} = \lambda \sum_{i=1}^{6} x_i N_{i,14} + N_{7,14}$$

$$\det(\mathtt{F}) = 0 \Rightarrow \det \begin{pmatrix} X_1 & X_2 & X_3 \\ X_4 & X_5 & X_6 \\ X_7 & X_8 & X_9 \end{pmatrix} = 0$$

This set of equations is equivalent to the initial system of polynomial equations but it is simpler because instead of eight quadratic and one cubic equation in 9 unknowns (assuming $f_9 = 1$) we have only 7 equations (six quadratic and one cubic) in 7 unknowns. We will use these 7 equations to create the action matrix for the polynomial $f = \lambda$.

3.2 Computing B and the number of solutions

To compute B, we solve our problem in a random finite prime field $\mathbb{Z}_p$ ($\mathbb{Z}/\langle p\rangle$) with $p >> 7$, where exact arithmetic can be used and numbers can be represented in a simple and efficient way. It speeds up computations and minimizes memory requirements.

We use algebraic geometric software Macaulay 2, which can compute in finite fields, to solve the polynomial equations for many random coefficients from $\mathbb{Z}_p$, to compute the number of solutions, the Gröbner basis, and the basis B. If the basis B remains stable for many different random coefficients, it is generically equivalent to the basis of the original system of polynomial equations. Then, we are done.

We can use the Gröbner basis and the basis B computed for random coefficients from $\mathbb{Z}_p$ thanks to the fact that in our class of problems the way of computing the Gröbner basis is always the same and for particular data these Gröbner bases differ only in coefficients. This holds for B, which consists of the same monomials, as well. Also, the way of obtaining polynomials that are necessary to create the action matrix is always the same and for a general data the generated polynomials differ again only in their coefficients. This way we have found that our problem has 16 solutions. To create the action matrix, we use the graded reverse lexicographic ordering with $x_1 > x_2 > x_3 > x_4 > x_5 > \lambda > x_6$. With this ordering, we get the basis $B = (x_6^3, \lambda^2, x_1x_6, x_2x_6, x_3x_6, x_4x_6, x_5x_6, x_6^2, x_1, x_2, x_3, x_4, x_5, \lambda, x_6, 1)$ of the algebra $A = \mathbb{C}[x_1, x_2, x_3, x_4, x_5, \lambda, x_6]/I$ which, as we shall see later, is suitable for finding the action matrix $\mathtt{M}_\lambda$.

3.2.1 Computing the number of solutions and basis B in Macaulay 2

Here we show the program for computing the number of solutions and basis B of our problem in Macaulay 2. Similar programs can be used to compute the number of solutions and basis of the algebra A for other problems.

```
// polynomial ring with coefficients form Z_p (Z_30097)
```

```
R = ZZ/30097[x_1..x_9, MonomialOrder=>Lex];
// Formulate the problem over Z_p => the set of equations
   eq
   (known variables -> random numbers from Z_p )
F = matrix({{x_1,x_2,x_3},{x_4,x_5,x_6},{x_7,x_8,1_R}});
X1 = matrix{apply(8,i->(random(R^2,R^1)))}
   ||matrix({{1_R,1_R,1_R,1_R,1_R,1_R,1_R,1_R}});
Z1 = matrix({{0_R,0_R,0_R,0_R,0_R,0_R,0_R,0_R}})
   ||matrix({{0_R,0_R,0_R,0_R,0_R,0_R,0_R,0_R}})
   || matrix{apply(8, i->X1_(0,i)^2+X1_(1,i)^2)};
X2 = matrix{apply(8,i->(random(R^2,R^1)))}
   ||matrix({{1_R,1_R,1_R,1_R,1_R,1_R,1_R,1_R}});
Z2 = matrix({{0_R,0_R,0_R,0_R,0_R,0_R,0_R,0_R}})
   ||matrix({{0_R,0_R,0_R,0_R,0_R,0_R,0_R,0_R}})
   || matrix{apply(8, i->X2_(0,i)^2+X2_(1,i)^2)};
P1 = X1 + x_9*Z1;
P2 = X2 + x_9*Z2;
eq = apply(8, i->(transpose(P1_[i]))*F*(P2_[i]));
// Ideal generated by polynomials eq + the polynomial
   det(F)
I1 = ideal(f) + ideal det F;
// Compute the number of solutions
gbTrace 3
dim I1  //the dimension of ideal I (zero-dimensional
   ideal <=> V(I) is a finite set
degree I1 //the number of solutions (the number of
   points in V(I))
// Groebner basis
transpose gens gb I1
 A = R/I1 //the quotient ring A=ZZ/30097[x_1..x_9] /I1
B = basis A //the basis of the quotient ring A
```

The above program in Macaulay 2 gives not only the number of solutions and the basis B of the algebra A , but also the information like how difficult it is to compute the Gröbner basis, and how many and which S-polynomials have to be generated.

The level of verbosity is controlled with the command `gbTrace(n)`.

- For n=0 no extra information is produced
- For n=3 string “m”, ”o” and “r” are printed.
  - The letter “m” is printed when a new S-pair does not reduce to zero and is added to the basis.
  - the letter ”o” indicates that an S-pair or generator is reduced to zero, but no new syzygy occurred.
  - and the letter ”r” when an S-pair has been removed
- For n=100 we get which S-polynomials were computed, from which polynomials were these S-polynomials created, which S-polynomial did not reduce to 0, which inserted into the basis and so on.

See [26] for more information about how to use Macaulay 2 to build minimal solvers.

3.3 Constructing action matrix

Here we construct the action matrix $\mathtt{M}_\lambda$ for multiplication by polynomial $f = \lambda$. The method described in section 2.2 calls for generating polynomials $q_i = \lambda \mathbf{x}^{\alpha(i)} + \sum_{j=1}^{N} c_{ji}\mathbf{x}^{\alpha(j)} \in I$.

In graded orderings, the leading monomials of q_i are $\lambda \mathbf{x}^{\alpha(i)}$. Therefore, to find q_i, it is enough to generate at least one polynomial in the required form for each leading monomial $\lambda \mathbf{x}^{\alpha(i)}$. This can be, for instance, done by systematically generating polynomials of I with ascending leading monomials and testing them. We stop when all necessary polynomials q_i are obtained. Let d be the degree of the highest degree polynomial from initial generators F. Then we can generate polynomials q_i from F in this way:

1. Generate all monomial multiples $\mathbf{x}^\alpha f_i$ of degree $\leq d$.
2. Write the polynomial equations in the form $\mathtt{M}X = 0$, where $\mathtt{M}$ is the coefficient matrix and X is the vector of all monomials ordered by the used monomial ordering.
3. Simplify matrix $\mathtt{M}$ by the Gauss-Jordan (G-J) elimination.
4. If all necessary polynomials q_i have been generated, stop.
5. If no new polynomials with degree $< d$ were generated by G-J elimination, set $d = d + 1$.
6. Go to 1.

In this way we can systematically generate all necessary polynomials. Unfortunately, we also generate many unnecessary polynomials. We use Macaulay 2 to identify the unnecessary polynomials and avoid generating them.

In the process of creating the action matrix $\mathtt{M}_\lambda$, we represent polynomials by rows of the matrix of their coefficients. Columns of this matrix are ordered according to the monomial ordering. The basic steps of generating the polynomials necessary for constructing the action matrix are as follows:

1. We begin with six 2^{nd} degree polynomials $f_1^{(0)}, \ldots, f_6^{(0)}$ and one 3^{rd} degree polynomial $f_7^{(0)} = \det(\mathtt{F}) = 0$. We perform G-J elimination of the matrix representing the six 2^{nd} degree polynomials and reconstruct the six reduced polynomials $f_1^{(1)}, \ldots, f_6^{(1)}$.

2. We multiply $f_1^{(1)}, \ldots, f_6^{(1)}$ by $1, x_1, x_2, x_3, x_4, x_5, \lambda,\ x_6$ and add $f_7^{(0)}$ to get 49 2^{nd} and 3^{rd} degree polynomials $f_1^{(2)}, \ldots, f_{49}^{(2)}$. They can be represented by 119 monomials and a 49×119 matrix with rank 49, which we simplify by one G-J elimination again.

3. We obtain 15 new 2^{nd} degree polynomials ($f_{29}^{(2)} \ldots, f_{43}^{(2)}$), six old 2^{nd} degree polynomials (reduced polynomials $f_1^{(1)}, \ldots, f_6^{(1)}$, now $f_{44}^{(2)} \ldots, f_{49}^{(2)}$) and 28 polynomials of degree three. In order to avoid adding 4^{th} degree polynomials on this stage we add only $x_1, x_2, x_3, x_4, x_5, \lambda, x_6$ multiples of these 15 new 2^{nd} degree polynomials to the polynomials $f_1^{(2)}, \ldots, f_{49}^{(2)}$. Thus obtaining 154 polynomials $f_1^{(3)}, \ldots, f_{154}^{(3)}$ representable by a 154×119 matrix, which has rank 99. We simplify it by G-J elimination and obtain $f_1^{(4)}, \ldots, f_{99}^{(4)}$.

4. The only 4^{th} degree polynomial that we need is a polynomial in the form $\lambda x_6^3 + h,\ h \in A$. To obtain this polynomial, we only need to add monomial multiples of one polynomial g from $f_1^{(4)}, ..., f_{99}^{(4)}$ which has leading monomial $LM(g) = \lambda\, x_6^2$. This is possible thanks to our monomial ordering. All polynomials $f_1^{(4)}, ..., f_{99}^{(4)}$ and $x_1, x_2, x_3, x_4, x_5, \lambda, x_6$ multiples of the 3^{rd} degree polynomial g with $LM(g) = \lambda\, x_6^2$ give 106 polynomials $f_1^{(5)}, ..., f_{106}^{(5)}$ which can be represented by a 106×126 matrix of rank 106. After another G-J elimination, we get 106 reduced polynomials $f_1^{(6)}, ..., f_{106}^{(6)}$. Because the polynomial g with $LM(g) = \lambda\, x_6^2$ has already been between the polynomials $f_1^{(2)}, ..., f_{49}^{(2)}$, we can add its monomial multiples already in the 3^{rd} step. After one G-J elimination we get the same 106 polynomials. In this way, we obtain polynomial q with $LM(q) = \lambda\, x_6^3$ as $q = \lambda\, x_6^3 + c_1 x_2 x_6^2 + c_2 x_3 x_6^2 + c_3 x_4 x_6^2 + c_4 x_5 x_6^2 + h'$, for some $c_1, ..., c_4 \in \mathbb{C}$ and $h' \in A$ instead of the desired $\lambda\, x_6^3 + h,\ h \in A$.

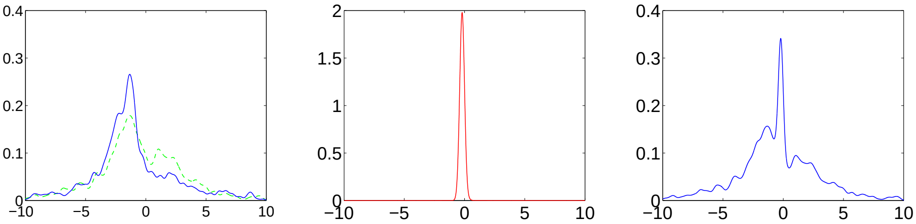


Figure 2: Distribution of real roots in $[-10, 10]$ using kernel voting for 500 noiseless point matches, 200 estimations and $\lambda_{true} = -0.2$. (Left) Parasitic roots (green) vs. roots for mismatches (blue). (Center) Genuine roots. (Right) All roots, 100% of inliers.

5. Among the polynomials $f_1^{(6)}, \ldots, f_{106}^{(6)}$, there are 12 out of the 14 polynomials that are required for constructing the action matrix. The first polynomial which is missing is the above mentioned polynomial $q_1 = \lambda\, x_6^3 + h_1, h_1 \in A$. To obtain this polynomial from q, we need to generate polynomials from the ideal with leading monomials $x_2x_6^2$, $x_3x_6^2$, $x_4x_6^2$, and $x_5x_6^2$. The second missing polynomial is $q_2 = \lambda^3 + h_2$, $h_2 \in A$. All these 3^{rd} degree polynomials from the ideal I can be, unfortunately, obtained only by eliminating the 4^{th} degree polynomials. To get these 4^{th} degree polynomials, the polynomial with leading monomial $x_1x_6^2$, resp. $x_2x_6^2$, $x_3x_6^2$, $x_4x_6^2$, $x_5x_6^2$ is multiplied by λ and subtracted from the polynomial with leading monomial λx_6 multiplied by x_1x_6, resp. by x_2x_6, x_3x_6, x_4x_6, x_5x_6. After G-J elimination, a polynomial with the leading monomial $x_2x_6^2$, resp. $x_3x_6^2$, $x_4x_6^2$, $x_5x_6^2$, λ^3 is obtained.

6. All polynomials needed for constructing the action matrix are obtained. Action matrix M_λ is constructed.

The pseudocode of the algorithm for creating the action matrix M_λ and computing $\lambda' s$ and $F' s$ from it is can be found Appendix A.

3.4 Solving equations using eigenvectors

The eigenvectors of $\mathtt{M}_\lambda$ give solutions for $x_1, x_2, x_3,\ x_4, x_5, \lambda, x_6$. Using a back-substitution, we obtain solutions for $f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, \lambda$. In this way we obtain 16 (complex) solutions. Generally less than 10 solutions are real.

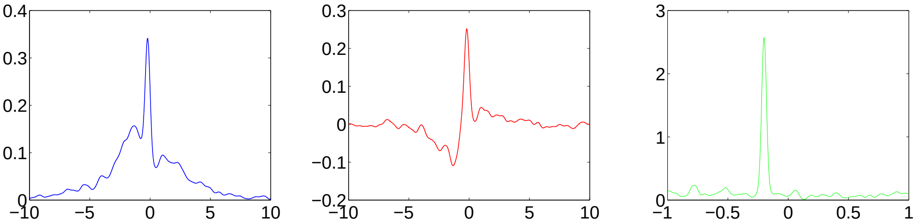


Figure 3: Distribution of real roots using kernel voting for 500 noiseless point matches, 100% inliers, 200 groups and $\lambda_{true} = -0.2$. (Left) Distribution of all roots in $[-10, 10]$. (Center) Distribution of all roots minus the distribution of roots from mismatches in $[-10, 10]$. (Right) Distribution of all roots in $[-1, 1]$.

4 Experiments

We test our algorithm on both synthetic (with various levels of noise, outliers and radial distortions) and real images and compare it to the existing 9-point algorithms for correcting radial distortion [6, 15]. We can get up to 16 complex roots. In general, more than one and less than 10 roots are real. If there is more than one real root, we need to select the best root, the root which is consistent with most measurements. To do so, we treat the real roots of the 16 (in general complex) roots obtained by solving the equations for one input as real roots from different inputs and use RANSAC [5, 11] or kernel voting [15] for several (many) inputs to select the best root among all generated roots. The kernel voting is done by a Gaussian kernel with fixed variance and the estimate of λ is found as the position of the largest peak. See [15] for more on kernel voting for this problem. To evaluate the performance of our algorithm, we distinguish three sets of roots.

- "All roots" is the set of all real roots obtained by solving the equations for K (different) inputs.
- "Genuine roots" denote the subset of all roots obtained by selecting the real root closest to the true λ for each input containing only correct matches. The set of genuine roots can be identified only in simulated experiments.
- "Parasitic roots" is the subset of all roots obtained by removing the genuine roots from all roots when everything is evaluated on inputs containing only correct matches.

The results of our experiments for the kernel voting are shown in figure 2. Figure 2 (Left) shows that the distribution of all real roots for mismatches is similar to the distribution of the parasitic roots. This allows to treat parasitic roots in the same way as the roots for mismatches. Figures 2 (Left and Center) show that the

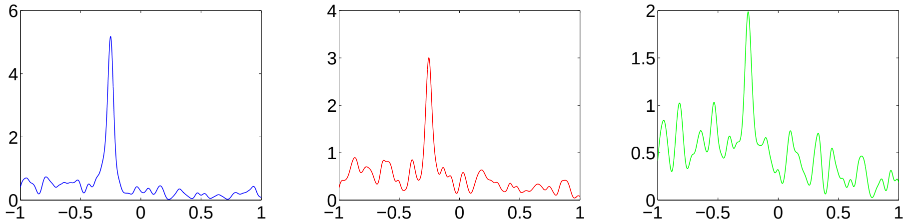


Figure 4: Kernel voting results, for $\lambda_{true} = -0.25$, noise level $\sigma = 0.2$ (1 pixel), image size 768×576 and (Left) 100% inliers, (Center) 90% inliers (Right) 80% inliers. Estimated radial distortion parameters were (Left) $\lambda = -0.2510$ (Center) $\lambda = -0.2546$ (Right) $\lambda = -0.2495$.

distribution of genuine roots is very sharp compared to the distribution of parasitic roots and roots for mismatches. Therefore, it is possible to estimate the true λ as the position of the largest peak, figure 2 (Right). These experiments show that it is suitable to use kernel voting and that it make sense to select the best root by casting votes from all computed roots. It is clear from results shown in figure 3 that it is meaningful to vote for λ's either (i) within the range where the most of the computed roots fall (in our case [-10,10]), figure 3 (Left), or (ii) within the smallest range in which we are sure that the ground truth lie (in our case [-1,1]), figure 3 (Right). For large number of input data, it might also makes sense to subtract the apriory computed distribution of all real roots for mismatches from the distribution of all roots.

4.1 Tests on synthetic images

We initially studied our algorithm using synthetic datasets. Our testing procedure was as follows:

1. Generate a 3D scene consisting of N $(= 500)$ random points distributed uniformly within a cuboid. Project $M\%$ of the points on image planes of the two displaced cameras. These are matches. In both image planes, generate $(100 - M)\%$ random points distributed uniformly in the image. These are mismatches. Altogether, they become undistorted correspondences.
2. Apply the radial distortion to the undistorted correspondences to generate noiseless distorted points.
3. Add Gaussian noise of standard deviation σ to the distorted points.
4. Repeat K times (We use $K = 100$ here, but in many cases K from 30 to 50 is sufficient).

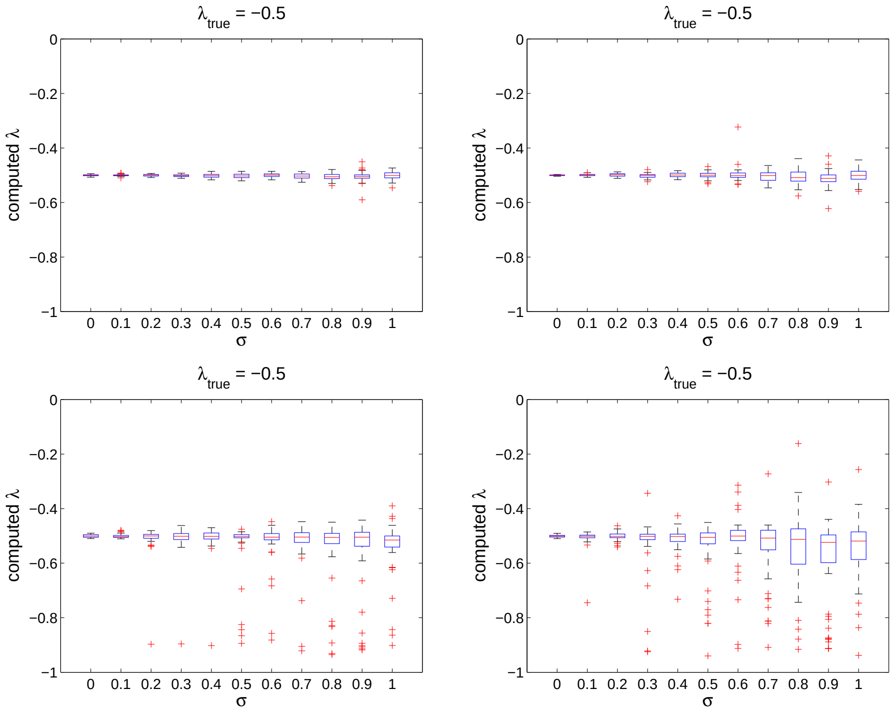


Figure 5: Estimated λ as a function of noise σ, ground truth $\lambda_{true} = -0.5$ and (Top) $inliers = 90\%$, (Bottom) $inliers = 80\%$. Blue boxes contain values from 25% to 75% quantile. (Left) 8-point algorithm. (Right) 9-point algorithm.

(a) Randomly choose 8 point correspondences from given N correspondences.

(b) Normalize image point coordinates to $[-1, 1]$ by subtracting the image center and dividing by $\max(image\_width/2, image\_height/2)$..

(c) Find up to 16 roots of the minimal solution to the autocalibration of radial distortion.

(d) Select the real roots in the feasible interval, e.g., $-1 < \lambda < 1$ and the corresponding $\mathtt{F}$'s.

5. Use kernel voting to select the best root.

The resulting density functions for different outlier contaminations and for the noise level 1 pixel are shown in figure 4. Here, $K = 100$, image size was 768×576 and $\lambda_{true} = -0.25$. In all cases, a good estimate, very close to the true λ, was found as the position of the maximum of the root density function. We conclude, that the method is robust to mismatches and noise.

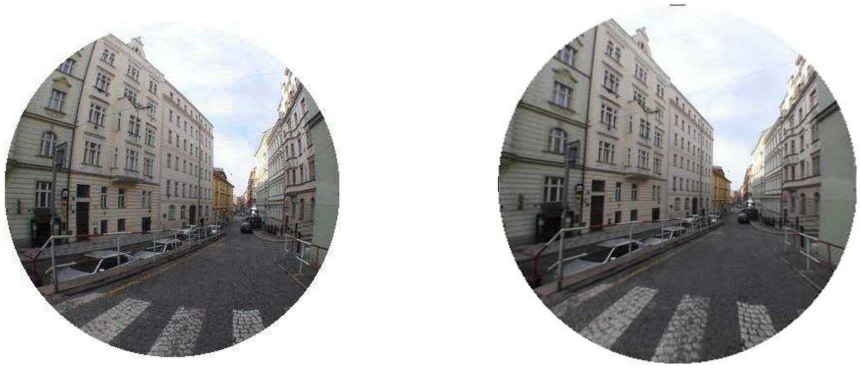


Figure 6: Real data. (Left) Input image with significant radial distortion. (Right) Corrected image.

In the next experiment we study the robustness of our algorithm to increasing levels of Gaussian noise added to the distorted points. We compare our results to the results of two existing 9-point algorithms [6, 15]. The ground truth radial distortion λ_{true} was -0.5 and the level of noise varied from $\sigma = 0$ to $\sigma = 1$, i.e. from 0 to 5 pixels. Noise level 5 pixels is relatively large but we get good results even for this noise level and 20% of outliers.

Figure 5 (Left) shows λ computed by our 8-point algorithm as a function of noise level σ. Fifty lambdas vere estimated from fifty 8-tuples of correspondences randomly drawn for each noise level and (Top) 90% and (Bottom) 80% of inliers. The results are presented by the Matlab function *boxplot* which shows values 25% to 75% quantile as a blue box with red horizontal line at median. The red crosses show data beyond 1.5 times the interquartile range. The results for 9-point algorithms [6, 15], which gave exactly identical results, are shown for the same input, figure 5 (Right).

The median values for 8-point as well as 9-point algorithms are very close to the ground truth value $\lambda_{true} = -0.5$ for all noise levels. The variances of the 9-point algorithms, 5 (Right), are considerably larger, especially for higher noise levels, than the variances of the 8-point algorithm 5 (Left). The 8-point algorithm thus produces more good estimates for the fixed number of samples. This is good both for RANSAC as well as for kernel voting.

4.2 Tests on real images

The input images with relatively large distortion, figures 1 (Left) and 6 (Left), were obtained as cutouts from $180°$ angle of view fish-eye images. Tentative

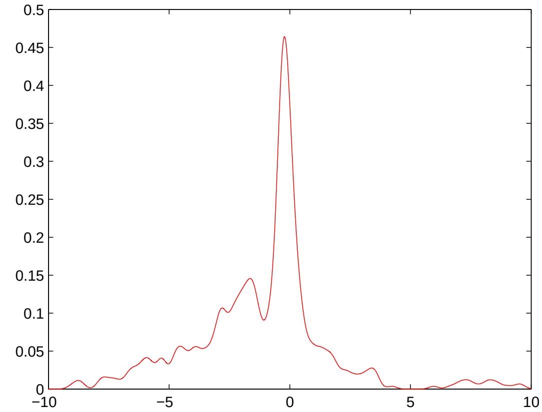


Figure 7: Distribution of real roots obtained by kernel voting for image in figure 6. Estimated $\lambda = -0.22$.

point correspondences were found by the wide base-line matching algorithm [17]. They contained correct as well as incorrect matches. Distortion parameter λ was estimated by our 8-point algorithm and the kernel voting method. The input (Left) and corrected (Right) images are presented in figures 1 and 6. Figure 7 shows the distribution of real roots, for image from figure 6 (Left), from which $\lambda = -0.22$ was estimated as the argument of the maximum.

A Appendix

Algorithm for correcting radial lens distortion from 8 point correspondences

Input: Eight corresponding pairs Q1, Q2.

Output: F = vector of computed 3x3 possible fundamental matrices
lambda = radial distortion parameter

function $[lamda, F] = EightPoint(Q1, Q2)$

Using Q1 and Q2 create matrix M (M.X=0)
$[U, S, V] = svd(M)$
$N = V(:, 9:15);$
$B = zeros(161, 330);$
Add six 2^{nd} degree equations ($f_1^{(0)}, \ldots, f_6^{(0)}$) to the matrix B
$B = ReduceSubmatrix(B);$
Multiply $f_1^{(1)}, \ldots, f_6^{(1)}$ (rows 1-6 of B) by $1, x_1, x_2, x_3, x_4, x_5, \lambda, x_6$
Add multiplied polynomials to B
Add equation $\det(F) = 0$ to B

$B = ReduceSubmatrix(B);$

Multiply 15 new 2^{nd} degree polynomials from B ($f_{29}^{(2)} \ldots, f_{43}^{(2)}$) (rows 29-43 of B) by $1, x_1, x_2, x_3, x_4, x_5, \lambda, x_6$

Multiply the 3^{rd} degree polynomial with LT $(\lambda.x_6^2)$ (row 28 of B) by

$1, x_1, x_2, x_3, x_4, x_5, \lambda, x_6$

Add multiplied polynomials to B

$B = ReduceSubmatrix(B);$

$p_1 = \lambda.q_1$, where q_1 is a polynomial with $LM\ (q_1) = (x_1 x_6^2)$ (row 84 of B)

$p_2 = x_1 x_6.q_2$, where q_2 is a polynomial with $LM\ (q_2) = (\lambda x_6)$ (row 106 of B)

Add polynomial $p_2 - p_1$ to B (as 85 row of B)

$B = ReduceRowInMatrix(B, 85);$

$p_1 = \lambda.q_1$, where q_1 is a polynomial with $LM\ (q_1) = (x_2 x_6^2)$ (row 85 of B)

$p_2 = x_2 x_6.q_2$, where q_2 is a polynomial with $LM\ (q_2) = (\lambda x_6)$ (row 107 of B)

Add polynomial $p_2 - p_1$ to B (as 86 row of B)

$B = ReduceRowInMatrix(B, 86);$

$p_1 = \lambda.q_1$, where q_1 is a polynomial with $LM\ (q_1) = (x_3 x_6^2)$ (row 86 of B)

$p_2 = x_3 x_6.q_2$, where q_2 is a polynomial with $LM\ (q_2) = (\lambda x_6)$ (row 108 of B)

Add polynomial $p_2 - p_1$ to B (as 87 row of B)

$B = ReduceRowInMatrix(B, 87);$

$p_1 = \lambda.q_1$, where q_1 is a polynomial with $LM\ (q_1) = (x_4 x_6^2)$ (row 87 of B)

$p_2 = x_4 x_6.q_2$, where q_2 is a polynomial with $LM\ (q_2) = (\lambda x_6)$ (row 109 of B)

Add polynomial $p_2 - p_1$ to B (as 88 row of B)

$B = ReduceRowInMatrix(B, 88);$

$p_1 = \lambda.q_1$, where q_1 is a polynomial with $LM\ (q_1) = (x_5 x_6^2)$ (row 88 of B)

$p_2 = x_5 x_6.q_2$, where q_2 is a polynomial with $LM\ (q_2) = (\lambda x_6)$ (row 110 of B)

Add polynomial $p_2 - p_1$ to B (as 63 row of B)

$B = ReduceRowInMatrix(B, 63);$

% create action matrix

$A(16, :) = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0];$

$A(15, :) = -B(111, [294, 315 : 320, 322 : 330]);$

```
A(14,:) = [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];
A(13 : -1 : 1,:) = -B([110, 109, 108, 107, 106, 90, 83, 82, 81, 80, 79, 63, 7],
[294, 315 : 320, 322 : 330]);
[V, D] = eig(A);
SOLS = V(9 : 15,:)./(ones(7, 1) * V(16,:));
I = find(not(imag(SOLS(6,:))));
lambda = SOLS(6, I);
x = SOLS([1, 2, 3, 4, 5, 7],:);
Fvec = N(1 : 9,:) * [x; ones(1, 16)];
Fvec = Fvec./(ones(9, 1) * sqrt(sum(Fvec.^2)));
I = find(not(imag(Fvec(1,:))));
Fvec = Fvec(:, I);
for i=1:size(Fvec,2)
    F{i} = reshape(Fvec(:,i),3,3)';
end
```

References

[1] C. Bräuer-Burchardt and K. Voss. A new algorithm to correct fish-eye and strong wide-angle-lens-distortion from single images. *ICIP 2001*, pp. 225–228.

[2] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Springer-Verlag, 2005.

[3] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms*. Springer-Verlag, 1992.

[4] D. Devernay and O. Faugeras. Straight lines have to be straight. *MVA*, 13(1):14–24, 2001.

[5] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981.

[6] A. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. *CVPR 2001*, pp. 125–132.

[7] D. Claus and A. Fitzgibbon. A rational function lens distortion model for general cameras. In *CVPR 2005*, volume 1, pages 213–219, 2005.

[8] J.-C. Faugere. A new efficient algorithm for computing gröbner bases (f_4). *Journal of Pure and Applied Algebra*, 139(1-3):61–88, 1999.

[9] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE PAMI*, 25(8):930–943, 2003.

[10] R. Hartley and S. Kang. Parameter-free radial distortion correction with centre of distortion estimation. *ICCV 2005*, pp. 1834–1841.

[11] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[12] S. Kang. Catadioptric self-calibration. *CVPR 2000*, pp.

[13] S. Kang. Radial distortion snakes. *IAPR MVA Workshop 2000*, pp. 603–606, Tokyo.

[14] H. Li. A simple solution to the six-point two-view focal-length problem. *ECCV 2006*, pp. 200–213.

[15] H. Li and R. Hartley. A non-iterative method for correcting lens distortion from nine-point correspondences. *OMNIVIS 2005*.

[16] H. Li and R. Hartley. Five-point motion estimation made easy. *ICPR 2006*, pp. 630–633.

[17] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[18] B. Micusik and T. Pajdla. Estimation of omnidirectional camera model from epipolar geometry. *CVPR 2003*, pp. 485–490.

[19] D. Nister. An efficient solution to the five-point relative pose. *IEEE PAMI*, 26(6):756–770, 2004.

[20] R. Steele and C. Jaynes. Overconstrained linear estimation of radial distortion and multi-view geometry. *ECCV 2006*, pp. 253–264.

[21] G. Stein. Lens distortion calibration using point correspondences. *CVPR 1997*, pp. 600:602.

[22] H. Stewenius. *Gröbner basis methods for minimal problems in computer vision*. PhD thesis, Lund University, 2005.

[23] H. Stewenius, C. Engels, and D. Nister. Recent developments on direct relative orientation. *ISPRS J. of Photogrammetry and Remote Sensing*, 60:284–294, 2006.

[24] H. Stewenius, D. Nister, F. Kahl, and F. Schaffalitzky. A minimal solution for relative pose with unknown focal length. In *CVPR 2005*, pp. 789–794.

[25] H. Stewenius, D. Nister, M. Oskarsson, and K. Astrom. Solutions to minimal generalized relative pose problems. *OMNIVIS 2005*.

[26] H. Stewenius. Exercises in Algebraic Solving of Minimal Problems, 2005 http://www.vis.uky.edu/˜dnister/Tutorials/algebraic_geometry_exercises.pdf

[27] R. Strand and E. Hayman. Correcting radial distortion by circle fitting. *BMVC 2005*.

[28] S. Thirthala and M. Pollefeys. Multi-view geometry of 1d radial cameras and its application to omnidirectional camera calibration. *ICCV 2005*, pp. 1539–1546.

[29] S. Thirthala and M. Pollefeys. The radial trifocal tensor: A tool for calibrating the radial distortion of wide-angle cameras. *CVPR 2005* pp. 321–328.

[30] R. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE J. of Robotics and Automation*, 3(4):323344, 1987.

[31] Z. Zhang. On the epipolar geometry between two images with lens distortion. In *ICPR 1996*.

[32] Z. Zhang. A flexible new technique for camera calibration. *IEEE PAMI*, 22(11):1330–1334, 2000.

CENTER FOR
MACHINE PERCEPTION

CZECH TECHNICAL
UNIVERSITY

RESEARCH REPORT

ISSN 1213-2365

Image matching insensitive to the change of camera orientations

Akihiko Torii and Tomáš Pajdla

{torii, pajdla} @ cmp.felk.cvut.cz

Center for Machine Perception, Department of Cybernetics
Faculty of Elec. Eng., Czech Technical University in Prague

CTU–CMP–2006–11

December 22, 2006

Available at
WritethefullURLofthepaperhere,ifavailableintheelectronicform.

The 1st author was supported by the EC project FP6-IST-027787 DIRAC. The 2nd author was supported by the STINT Foundation under project Dur IG2003-2 062 and by the Czech Academy of Sciences under project 1ET101210406 and by the EC project MRTN-CT-2004-005439 VISIONTRAIN and by The under project HS. Any opinions expressed in this paper do not necessarily reflect the views of the European Community. The Community is not liable for any use that may be made of the information contained herein.

Research Reports of CMP, Czech Technical University in Prague, No. 11, 2006

Published by

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Image matching insensitive to the change of camera orientations

Akihiko Torii and Tomáš Pajdla

December 22, 2006

Abstract

We study the geometry of the projection with the aim to facilitate stereo-matching of images acquired by the NIKON fisheye $180°$ field-of-view lens camera. For the stereo matching, we assume that our target objects are locally planar. First, we formulate the homography for spherical cameras since all central omni-directional cameras can be converged to a spherical camera model. Second, we analyze the mapping between two omni-directional images through numerical experiments. Then, we consider the most suitable image representation of omni-directional images to obtain stereo-correspondences.

1 Introduction

Image matching plays an important role in object detection, tracking and visual event analysis. A classical approach [] to perspective image matching uses affine covariant features which can be repeatably detected and which provide stable descriptors.

Omnidirectional images with $180°$ field of view yield very non-linear image projections. The non-linearity often differs in the center and at the border of the image. Thus, a projection of an object to the center of the image may be related to the projection of the same object at the border of the image by a mapping that is far from an affine transform. This is unppleasant for two reasons. Firstly, some affinely covariant detectors such as [] will not yield image features related by the image transform induced by the camera pose change. Secondly, even if the detector was covariant w.r.t. a non-linear image transform, such as MSER [] detector, the affine invariant descriptors would be inapropriate.

In this work we study image transforms that alleviate problems of varying non-linearity of image projection. We address both issues raised above. First, we show a global image transform that makes the varying non-linearity to be closer to an afine trnasform and secondly we look at local rectification of images close to detected features.

In this study, we establish the stereo matching method which is insensitive to the change of orientation of central omni-directional cameras. Changes of viewpoint, i.e. translation and rotation of the viewpoint, induce changes in images. In a central omni-directional camera system, the changes induced by the rotation mainly depend on image formation and are fully compensated by back-projecting the image on a sphere whose center is identical to the viewpoint. It is possible to map any image obtained by a central camera to the image on a sphere if approximate knowledge of the image formation is available. For example, knowing the viewing angle is enough for the perspective image and the viewing angle and the lens or mirror shape are enough for the omni-directional image.

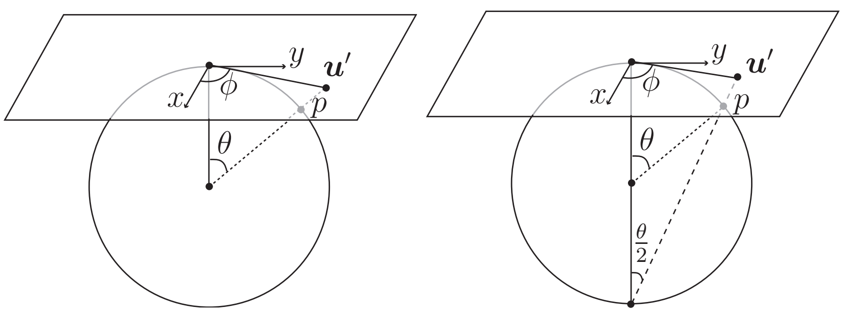
For reconstructing complete scenes, it is necessary to use many perspective cameras. Using omni-directional cameras calls for using fewer cameras compared to using traditional perspective camera systems. For the three-dimensional reconstruction using multiple (more than three) cameras, we are required to estimate the camera positions and viewing orientations (translation and rotation). If we reconstruct the same size scene, omni-directional camera systems need fewer estimations for camera positions and orientations compared to perspective camera systems. However, it is a problem that omni-directional cameras acquire distorted and low-resolution images since these observes large field of view. Therefore, it is necessary to calibrate the catadioptric and dioptric (shapes of mirrors and dioptric angles) of omni-directional camera systems carefully [7].

If the parameters of a central omni-directional camera system are known, all the omni-directional images can be transformed in the images onto a sphere in so called spherical images. The geometric analysis of central omni-directional system can be uniformly studied on the spherical camera systems. In principle, all image processing can be applied to images on a sphere but it must be re-defined on an irregular graph to compute image features on it. And the spherical geometry must be used as proposed in [3]. In this work, we show the geometrical transform of the planar objects, under the change of the viewpoint by the rotation and translation, observed on the images on a sphere.

Since the stereographic projection transforms the image on a sphere to the image on a plane identically as illustrated in Fig. 1 (a), the transformed stereographic image is suitable for representing the image for the application of standard image processing techniques.

Under camera rotations, regions in the stereographic image are mapped conformally such that circles map to circles since stereographic transform is a conformal mapping [2]. This circle preserving property is important to describe regions in the polar coordinates since the image is deformed along circles after normalization process (removing the scale and rotation) from the corresponding regions suitable for the wide baseline stereo matching [6]. In practical stereo matching of omni-directional images, most of planar regions in images can be related by affine transform. We show that pure rotation of omnidirectional camera induces image transform that can be well approximated by a similarity for planer regions of size useful for wide-baseline-stereo matching.

Furthermore, assuming that the regions extracted in image [6] are not too large, we obtain the centroid in a small region. In stereographic images, the deformation of corresponding small regions induced by rotation is eliminated if we rotate back the small region to the center of stereographic image with respect to the centroid. We call this stereographic image, which is reduced the scaling for rotation, *semi-scaled stereographic image*. We show that this semi-scaled stereographic image also improves the stereo-matching of two omni-directional images induced by translation.



(a) (b)

Figure 1: Stereographic projections of a sphere.

2 Omni-directional image representation for stereo matching

It is possible to transform central omni-directional images to images on a sphere [7]. In this study, we use a fish-eye lens camera as an omni-directional camera. Set $\mathbf{u} = (u, v)^\top$ be a point on the fish-eye camera image and $\mathbf{p} = (\theta, \varphi)$ be the corresponding point on a unit sphere expressed in polar coordinates. The mapping from fish-eye camera image to the spherical image is expressed by the following relation, –*Spherical transform* (transform from fish-eye image to a unit sphere)

$$\mathbf{p} = \begin{pmatrix} \theta \\ \varphi \end{pmatrix} = \begin{pmatrix} a||\mathbf{u}||/(1+b||\mathbf{u}||^2) \\ \tan^{-1}(u/v) \end{pmatrix} \tag{1}$$

where a and b are the parameters of the fish-eye camera (see [7]).

For employing standard image-processing techniques to omni-directional camera systems, a finite image on a plane is necessary. In practice, a fish-eye camera observes a hemispherical region. Therefore, the stereographic projection enables us to transform the image on a sphere to the finite image on a plane identically, –*Stereographic transform*

$$\mathbf{u}' = \begin{pmatrix} u' \\ v' \end{pmatrix} = 2\tan(\theta/2) \begin{pmatrix} \cos\varphi \\ \sin\varphi \end{pmatrix}, \tag{2}$$

where $\mathbf{u}' = (u', v')^\top$ is a point on the stereographic image and $\mathbf{p} = (\theta, \varphi)^\top$ is a point on a unit sphere expressed in polar coordinates. If the point on a unit sphere is expressed as $\mathbf{x} = (x, y, z)^\top$ in Euclidean coordinates, the stereographic projection is expressed by the equation,

$$\mathbf{u}' = \begin{pmatrix} u' \\ v' \end{pmatrix} = \frac{2}{1+z} \begin{pmatrix} x \\ y \end{pmatrix}, \tag{3}$$

See Figure 1.

On the other hand, we can transform a spherical image to perspective images. The perspective projection of point $\mathbf{x}$ on a sphere to a point $\mathbf{u}'$ on the tangential plane is expressed by the equation,

$$\mathbf{u}' = \begin{pmatrix} u' \\ v' \end{pmatrix} = \frac{1}{z} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{4}$$

We mention that, in the perspective back projection, it is not possible to map a whole spherical image onto a finite plane. In practice, it is necessary to project on some tangent planes of the sphere.

3 Theoretical analysis of images of planar objects for spherical camera

Geometrical transform of planar objects observed by two spherical cameras is described in this section. In the pin-hole camera systems, points in the two images of a plane are related by homography [1]. Since the projection model of omni-directional camera systems are expressed by a non-linear mapping, the mapping of point-to-point is not expressed in the exactly same way as for the pin-hole camera systems. We establish homography on spherical cameras since central omni-directional cameras can be converged to a spherical camera model. This point-to-point mapping between the spherical images for a planar object is the fundamental tool for analyzing the stereo matching of omni-directional images.

3.1 Transform points on a plane among spherical images

Setting the center of a spherical camera $\mathbf{C}$, say first camera, to be located at the origin of the world coordinate system, the center of a spherical camera $\mathbf{C}'$, say second camera, is shifted from the origin of the world coordinate system by the translation $\mathbf{t}$ without rotation. These spherical cameras observe a point $\mathbf{X}$ on a plane,

$$\Pi : \mathbf{v}^\top \mathbf{X} - d = 0, \tag{5}$$

where $\mathbf{v}$ is the normal vector of this plane and d is the distance among this plane and the origin of the world coordinate system. The point $\mathbf{X}$ is mapped to points

$$\mathbf{x} = \frac{\mathbf{X}}{|\mathbf{X}|}, \tag{6}$$

$$\mathbf{x}' = \frac{\mathbf{X} - \mathbf{t}}{|\mathbf{X} - \mathbf{t}|}, \tag{7}$$

on the spherical images, respectively. Assuming that the spherical images are on the unit sphere, the relation $|\mathbf{x}| = |\mathbf{x}'| = 1$ holds. From Eqs. (5) and (6), we obtain the equation,

$$\mathbf{X} = \frac{d}{\mathbf{v}^\top \mathbf{x}} \mathbf{x}. \tag{8}$$

Substitution of the Eq. (8) to Eq. (7) provides the point-to-point mapping:

$$\mathbf{x}' = \frac{d\mathbf{x} - (\mathbf{v}^\top \mathbf{x})\mathbf{t}}{|d\mathbf{x} - (\mathbf{v}^\top \mathbf{x})\mathbf{t}|}. \tag{9}$$

More generally, $\mathbf{x}'$ is measured in a different coordinate system, which is related to the coordinate system in the first camera, by a matrix A.

$$\mathrm{A}\mathbf{x}' = \frac{d\mathbf{x} - (\mathbf{v}^\top \mathbf{x})\mathbf{t}}{|d\mathbf{x} - (\mathbf{v}^\top \mathbf{x})\mathbf{t}|}. \tag{10}$$

Using the same camera, or, in other words, any Cartesian spherical camera, A becomes a rotation. Therefore, the mapping of the spherical images is expressed as

$$\mathbf{x}' = \mathrm{R}\frac{d\mathbf{x} - (\mathbf{v}^\top \mathbf{x})\mathbf{t}}{|d\mathbf{x} - (\mathbf{v}^\top \mathbf{x})\mathbf{t}|}. \tag{11}$$

See Figure 2 (a).

3.2 Equal Observation Angle (EOA) Transform

We would like to establish the approximation of homography on spherical images. We set the first camera $\mathbf{C}$ to the origin of the world coordinate system and the second camera $\mathbf{C}'$ shift by translation $\mathbf{t} = (0, 0, t)^\top$ without rotation. The points on the spherical images of $\mathbf{C}$ and $\mathbf{C}'$ are expressed as

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \cos\varphi \sin\theta \\ \sin\varphi \sin\theta \\ \cos\theta \end{pmatrix}, \tag{12}$$

$$\mathbf{x}' = \begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{pmatrix} \cos\varphi' \sin\theta' \\ \sin\varphi' \sin\theta' \\ \cos\theta' \end{pmatrix}, \tag{13}$$

where $0 \leq \theta, \theta' < \pi$ and $0 \leq \varphi, \varphi' < 2\pi$. Since we assumed no rotation between the coordinate systems of $\mathbf{C}$ and $\mathbf{C}'$,

$$\varphi = \varphi', \tag{14}$$

$$2\gamma = \theta' - \theta, \tag{15}$$

where γ is the bisector angle of $\angle\mathbf{CXC}'$. From Eqs. (14) and (15), we obtain the equations,

$$\sin\varphi' = \sin\varphi \cos 2\gamma + \sin 2\gamma \cos\varphi, \tag{16}$$

$$\cos\varphi' = \cos\varphi \cos 2\gamma - \sin 2\gamma \sin\varphi. \tag{17}$$

Substituting Eqs. (16) and (17) to Eq. (13), we have the relation,

$$\mathbf{x}' = (\cos(2\gamma)\mathbf{x} - \sin(2\gamma)\mathbf{y}), \tag{18}$$

where

$$\mathbf{y} = \begin{pmatrix} (x_1x_3)/\sqrt{1 - x_3^2} \\ (x_2x_3)/\sqrt{1 - x_3^2} \\ -\sqrt{1 - x_3^2} \end{pmatrix} \approx \begin{pmatrix} x_1x_3 \\ x_2x_3 \\ 1 \end{pmatrix}. \tag{19}$$

More generally, $\mathbf{x}'$ is measured in a different coordinate system, which is related to the coordinate system in the first camera, expressed by a rotation matrix R.

$$\mathbf{x}' = R(\cos(2\gamma)\mathbf{x} - \sin(2\gamma)\mathbf{y}), \tag{20}$$

We call this mapping from $\mathbf{x}$ to $\mathbf{x}'$ *Equal Observation Angle (EOA) Transform*. If a point of the first camera is on a plane and the normal vector of the plane and the bisector line are almost equivalent, the transform of points on the small region on the plane (illustrated as gray-colored region in Figure 2 (b)) is approximated by this EOA transform with the parameter γ.

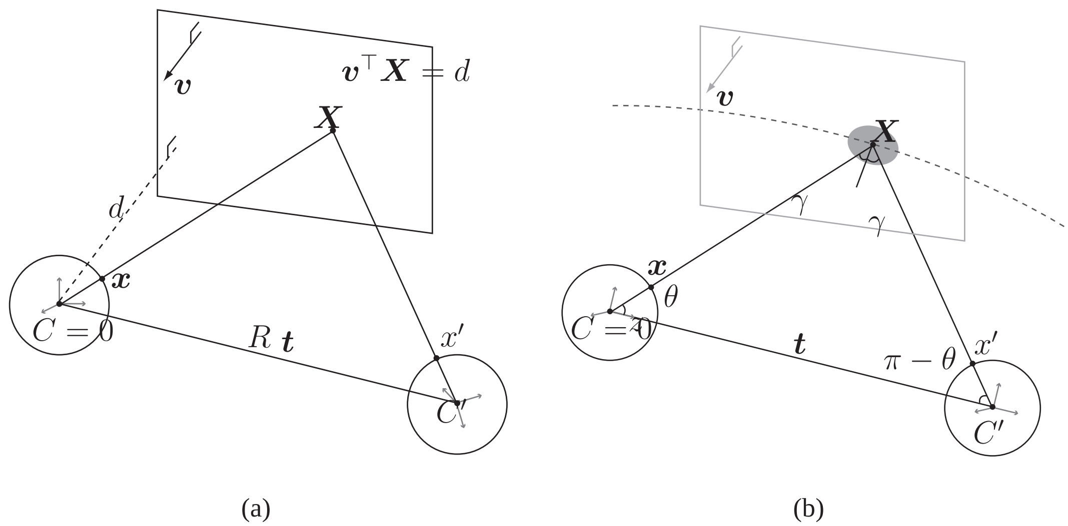


Figure 2: (a) General Plane Transform between spherical cameras. (b) Equal-Observation-Angle Transform between spherical cameras.

3.3 Analysis of scaling on fisheye-to-stereographic mapping

We formulate the mapping from a fisheye image onto a stereographic image. Let $\boldsymbol{u} = (u, v)^\top \in \mathbb{R}^2$ and $\boldsymbol{u}' = (u', v')^\top \in \mathbb{R}^2$ be the points on the fisheye and stereographic images, respectively. Setting $\boldsymbol{p} = (\theta, \varphi)^\top$ for $0 \leq \theta < \pi$ and $0 \leq \varphi \leq 2\pi$ to be a point on a unit sphere S^2, the mapping $h : \boldsymbol{u} \mapsto \boldsymbol{u}'$ is expressed in two steps such that $f : \boldsymbol{u} \mapsto \boldsymbol{p}$ and $g : \boldsymbol{p} \mapsto \boldsymbol{u}'$. In the one-parametric equidistant projection, the mapping f, which is the reverse mapping of the equidistant projection, is expressed as

$$\boldsymbol{p} = \begin{pmatrix} \theta \\ \varphi \end{pmatrix} = \begin{pmatrix} a\|\boldsymbol{u}\| \\ \tan^{-1}(\frac{u}{v}) \end{pmatrix}, \tag{21}$$

where a is a real positive constant which determines the scaling ratio according to the equidistant projection. The stereographic projection g is formulated as

$$\boldsymbol{u}' = g(\boldsymbol{p}) = \frac{2\tan(\frac{\theta}{2})}{b} \begin{pmatrix} \cos\varphi \\ \sin\varphi \end{pmatrix},$$

where b is a real positive constant which determines the scaling ratio according to the stereographic projection. Then, the mapping $h : \boldsymbol{u} \mapsto \boldsymbol{u}'$ is

$$\boldsymbol{u}' = g(f(\boldsymbol{u})) = \frac{\tan(a'\|\boldsymbol{u}\|)}{b'\|\boldsymbol{u}\|}\ \boldsymbol{u},$$

where $a' = a/2$ and $b' = b/2$. The change of the local area between $\boldsymbol{u}$ and $\boldsymbol{u}'$ is computed from the determinant of Jacobian J. The Jacobian is

$$J = \begin{pmatrix} \frac{\partial u'}{\partial u} & \frac{\partial u'}{\partial v} \\ \frac{\partial v'}{\partial u} & \frac{\partial v'}{\partial v} \end{pmatrix}.$$

The Jacobian determinant $|J|$ is

$$|J| = \frac{a'}{b'^2\|\boldsymbol{u}\|}\frac{|\sin(a'\|\boldsymbol{u}\|)|}{|\cos^3(a'\|\boldsymbol{u}\|)|}.$$

Since for $0 \le \theta < \pi$ $\sin(a'\|\boldsymbol{u}\|) \ge 0$ and $\cos^3(a'\|\boldsymbol{u}\|) \ge 0$, then

$$|J| = \frac{a'}{b'^2\|\boldsymbol{u}\|}\frac{\sin(a'\|\boldsymbol{u}\|)}{\cos^3(a'\|\boldsymbol{u}\|)}. \tag{22}$$

Hereafter, we derive the scaling ratio of the mapping h that satisfies the condition $|J| <= 1$. We set R and R' are the maximum radii of the fisheye and stereographic images. Setting $\theta_{\max}$ to be the maximum angle of the equidistant projection, $a = \theta_{\max}/R$ and $b = \frac{2\tan(\frac{\theta_{\max}}{2})}{R'}$. At the point of image center, i.e. $|\boldsymbol{u}| = 0$, L'Hospital's rule leads that

$$|J|_{|\boldsymbol{u}\|\to 0} = \frac{a'^2\cos(a'\|\boldsymbol{u}\|)}{b'^2(\cos^3(a'\|\boldsymbol{u}\|) - 3a'\|\boldsymbol{u}\|\cos^2(a'\|\boldsymbol{u}\|)\sin(a'\|\boldsymbol{u}\|))} = (\frac{a'}{b'})^2 = (\frac{a}{b})^2.$$

Therefore, we have

$$|J|_{|\boldsymbol{u}\|\to 0} = (\frac{R'}{R}\frac{\theta_{\max}}{2\tan(\frac{\theta_{\max}}{2})})^2.$$

For instance, when $\theta_{\max} = 95^\circ$ and $R = 600$, $R' \le 789.8194$ to satisfy $|J| \le 1$. Finally, it is possible to compute R' to satisfy the inequality

$$|J| = \frac{R'}{R}\frac{\theta_{\max}}{2\tan(\frac{\theta_{\max}}{2})}\frac{\sin(\frac{\theta_{\max}\|\boldsymbol{u}\|}{R})}{\cos(\frac{\theta_{\max}\|\boldsymbol{u}\|}{R})} \le 1.$$

4 Experimental Analysis

We use the algorithm for extracting affine invariant characterization of DR [5, 4].

1. compute first and second order statistics of MR
2. transform MR to have a unit covariance matrix
3. express data of normalized MR in polar coordinates,

4. apply one-dimensional FFT along θ-axis and keep only magnitude of complex numbers,

5. combine coefficients along r-axis according to polynomial $P_k(r)$.

This algorithm gives us a measurement vector for each DR. Using the measurement vectors, we establish tentative correspondence [5, 4].

In this study, for the original fish-eye image and the transformed stereographic images, we would like to observe how the objects on a plane in the scene are related for the transforms to polar coordinates if we assume the affine and similarity transform, respectively.

Figures 7 and 8 show the enlarged parts which include a planar object (target: a black circle on white paper) in the rotational and translational sequences, respectively.

In Figure 7 (a) (also (i)), the normal vector of the plane (target) is almost identical to the optical axis of spherical camera, i.e. the center of target is at the center of the image. (See Figure 4 (b).) In Figure 7 (b) (also (j)), the normal vector is directed perpendicular to the optical axis due to the pure rotation of the camera. Since the target is on the edge of the image, it is significantly affected by the distortion of fish-eye lens. If stereo correspondences are yielded by a transform between these two images, we can obtain the stereo correspondences from any two rotational sequences. In Figure 7, (e) and (f) ((m) and (n)) are the stereographic images transformed from (a) and (b) ((o) and (p)), respectively. Furthermore, in Figure 7, (c), (d), (g) and (h) are the polar patches must be explored from (a), (b), (e) and (f) based on similarity transform, respectively, and (k), (l), (o), and (p) are the polar patches obtained from (i), (j), (m), and (n) based on affine transform. For the two original images related by pure rotation, the polar patches rectified by a full affine transform yield stereo correspondences but those based on similarity do not. For the two stereographic images related by pure rotation, both of the polar patches rectified by a similarity and an affine transform yield stereo correspondences, since, on the stereographic images, circles on a plane are always observed as a circles (not ellipses).

In Figure 8 (a) (also (i)), the normal vector of the plane (target) is almost identity to the optical axis of spherical camera. In Figure 8 (b) (also (j)), the normal vector is directed parallel to the optical axis but the center of target is shifted due to the pure translation of the camera. (See Figure 4 (d).) Since the target is shifted to the edge of the image, this target is significantly affected by the distortion of fish-eye lens. In Figure 7, (e) and (f) ((m) and (n)) are the stereographic images transformed from (a) and (b) ((o) and (p)), respectively. Furthermore, in Figure 7, (c), (d), (g) and (h) are the polar patches obtained from (a), (b), (e) and (f) based on similarity transform, respectively, and (k), (l), (o), and (p) are the polar patches obtained from (i), (j), (m), and (n) based on affine transform. For the two original images and their stereographic images related by pure translation, the polar patches rectified by a full affine transform yield stereo correspondences but those based on similarity do not. In this case, circles on a plane are not mapped as a circles.

If the black colored circle is printed on center of a piece of square paper, the white colored region in the polar patches is symmetrical. However, in this experiment, since the black colored circle is printed on a piece of rectangle paper, the white colored region in the polar patches is not symmetrical.

These observations leads to the following conclusion. For the two fish-eye camera images obtained by pure rotation, it is possible to express the transform of planar objects using similarity if the original images are transformed to stereographic images. For the images obtained by pure translation, it is necessary to use full affine transform. However, we have to mention that affine transform "free" may be still unnecessarily.

4.1 Feature representation of extracted region

We denote $(\lambda_1^i, \lambda_2^i)^\top$ to be the square root of eigenvalues of covariance matrix related to the ith distinguished region (say DR). Assuming that $0 \leq \lambda_2^i \leq \lambda_1^i$, DR will be the *interesting region* as illustrated in Figure 15 if the eigenvalues satisfy that,

- $\lambda_{min} \leq \lambda_2^i$: λ_{min} is related to the discritization sampling and filtering of images. $\lambda_{min}(\delta, \Delta)$ is defined by $\Delta \leq \delta$ to avoid the aliasing. See Figure 15.

- $\lambda_1^i \leq \lambda_{max}$: the view field is practically limited.

Here, we focus on DRs obtained from a steregraphic image. We would like to reduce the ditortion caused by pure rotation applying geometric transform to each DR.

We assume that, before pure rotation, as illustrated in Figure 18 (a), we have a circle on a unit sphere and the centre of the circle corresponds to the pole on the unit sphere. We denote the radius of the circle as α using the angle. By the stereographic projection from the north pole to south pole, the circle on the unit sphere is mapped as c_0 onto the stereographic plane which is the tangential plane at the south pole. We set the radius of the mapped circle c_0 as r_o. We have the relation between the radius α and r_o,

$$\tan(\alpha/2) = r_0/2. \tag{23}$$

After the pure rotation, that is, the surface of the unit sphere is rotated, as illustrated in Figure 18 (b), it is possible to express the rotation as β on the plane which passes through the pole of the unit sphere and center of the circle on the unit sphere. After this rotation, the circle on the unit sphere is mapped onto the stereographic plane as c. We set the radius of c as r and the distance d between the center of image and the centroid of the circle r in the stereographic image. We have the relation among the radius r, the distance d, the angle α of the circle on the unit sphere and the angle of rotation β,

$$r + d = 2\tan(\alpha/2 + \beta/2), \tag{24}$$

$$\tan(\beta/2) = d/2. \tag{25}$$

Using Eqs. (23) to (24), we have the equation,

$$r_0 = \frac{2^2 r}{2^2 + d^2 + rd}. \tag{26}$$

For ellipses on a stereographic image, we apply this transform to reduce the distortion induced by pure rotation as approximation. For a DR, we have the singular values $\lambda = (\lambda_1, \lambda_2)^\top$ (assume $\lambda_1 \geq \lambda_2$), which corresponds to the major axis and minor axis of ellipse, and the distance d which is the distance between the centroid of the DR and origin of the stereographic image.

We define $\sigma = (\sigma_1, \sigma_2)$

$$\sigma_1 = \frac{2^2 \lambda_1}{2^2 + d^2 + \lambda_1 d}, \qquad \sigma_2 = \frac{2^2 \lambda_2}{2^2 + d^2 + \lambda_2 d}, \tag{27}$$

Figure 15 (b) illustrates the graph on which σ_1 and σ_2 are mapped. If (σ_1, σ_2) is not included in the colored region in the graph, the distinguished region is too small or too large to use for stereo matching.

Figure 19 (a) and (b) show σ_1 and σ_2 of all distinguished regions of the first original and stereographic image in Fig. 5. Figure 19 (c) and (d) show λ_1 and λ_2 of all distinguished regions of the first original and stereographic image in Fig. 5.

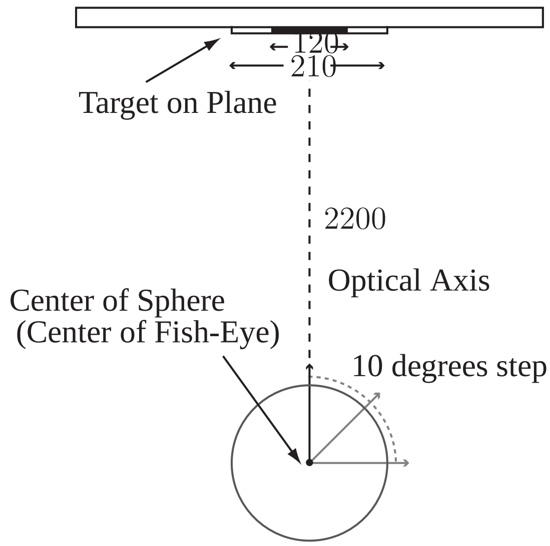
Figure 20 (a) and (b) show σ_1 and σ_2 of all distinguished regions of the sixth original and stereographic image in Fig. 5. Figure 20 (c) and (d) show λ_1 and λ_2 of all distinguished regions of the sixth original and stereographic image in Fig. 5.

Figure 21 (a) and (b) show σ_1 and σ_2 of all distinguished regions of the first original and stereographic image in Fig. 6. Figure 21 (c) and (d) show λ_1 and λ_2 of all distinguished regions of the first original and stereographic image in Fig. 6.

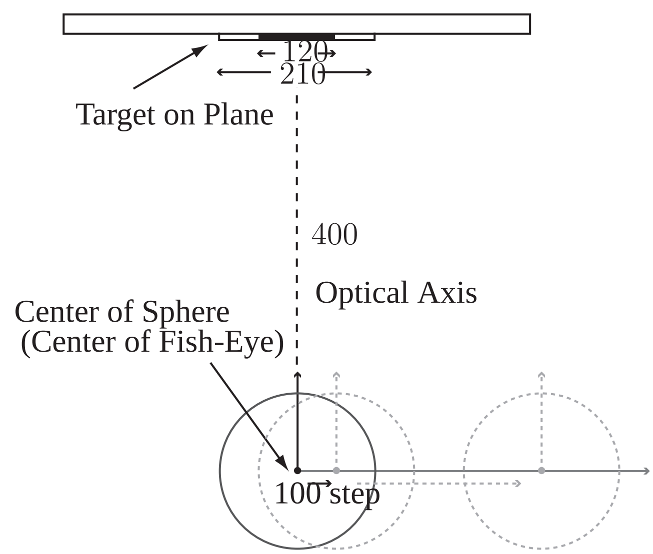
Figure 22 (a) and (b) show σ_1 and σ_2 of all distinguished regions of the eighth original and stereographic image in Fig. 6. Figure 22 (c) and (d) show λ_1 and λ_2 of all distinguished regions of the eighth original and stereographic image in Fig. 6.

(a) (b) (c)

Figure 3: Fisheye camera.



(a)



(b)

Figure 4: Experimental settings.

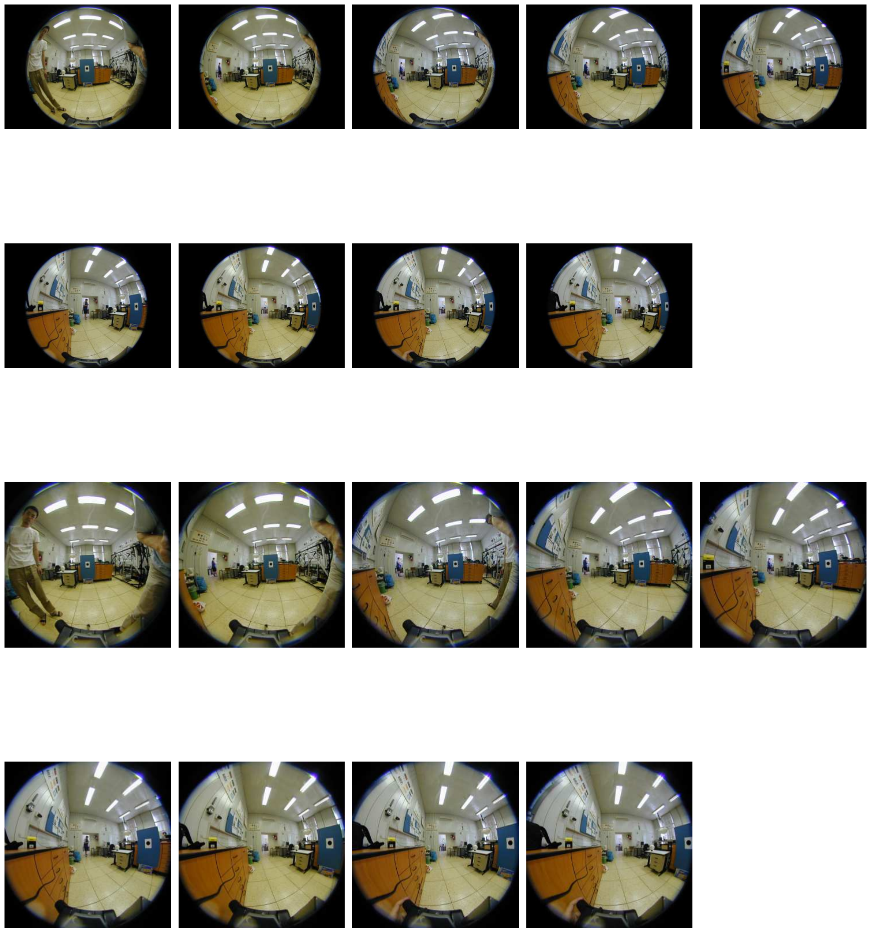


Figure 5: Sequences of images captured by pure rotation. Top tow row shows the sequence of original images. Bottom tow row shows the sequence of stereographic images.

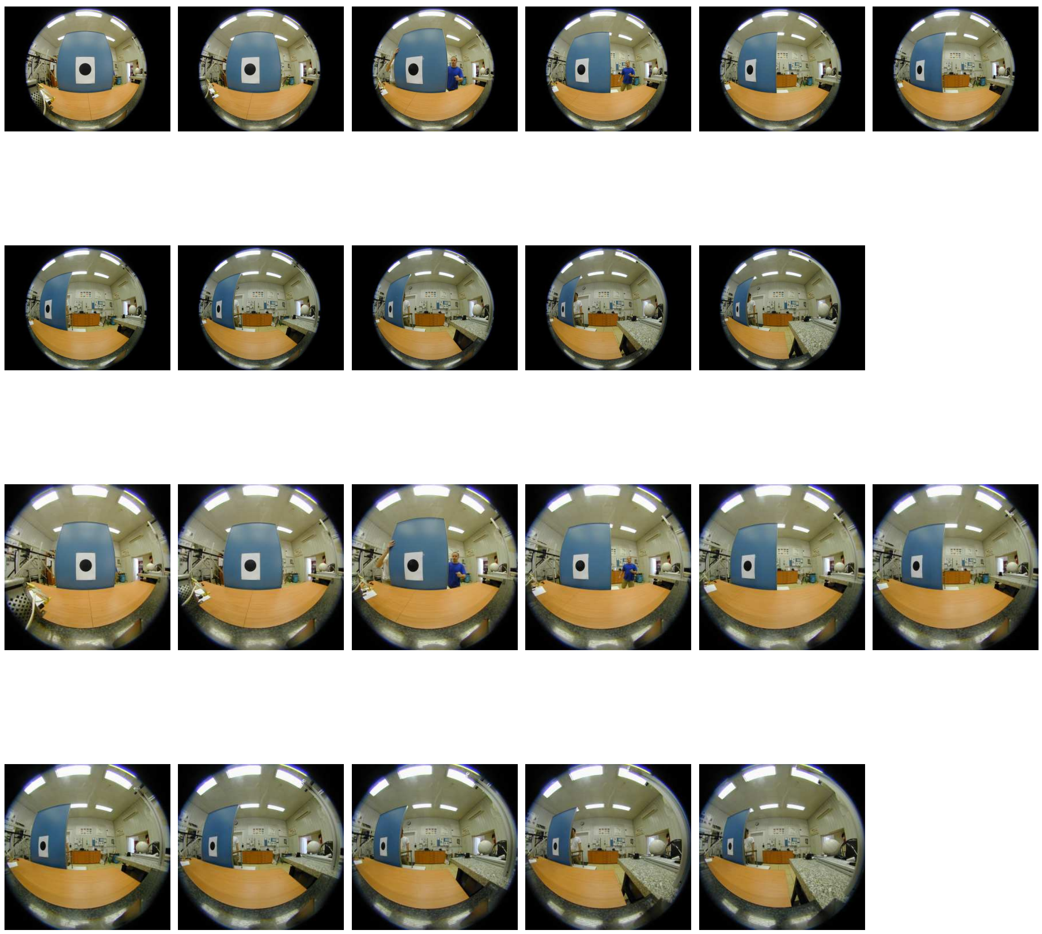


Figure 6: Sequences of images captured by pure translation. Top tow row shows the sequence of original images. Bottom tow row shows the sequence of stereographic images.

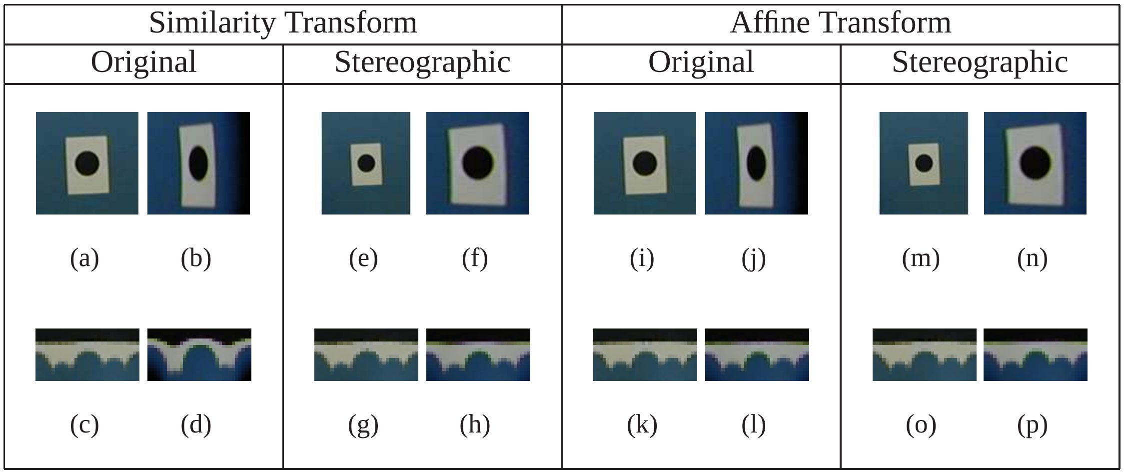


Figure 7: Enlarged parts of rotational images and their polar patches. (a) to (d): original images and their polar patches estimated by similarity transform. (e) to (h): stereographic images and their polar patches estimated by similarity transform. (i) to (l): original images and their polar patches estimated by affine transform. (e) to (h): stereographic images and their polar patches estimated by affine transform.

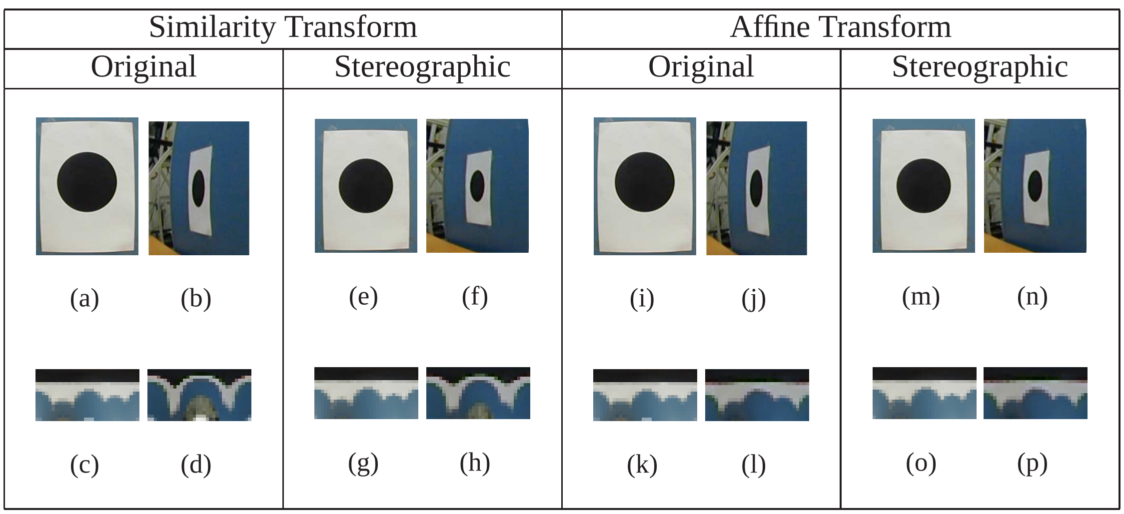


Figure 8: Enlarged parts of translational images and their polar patches. (a) to (d): original images and their polar patches estimated by similarity transform. (e) to (h): stereographic images and their polar patches estimated by similarity transform. (i) to (l): original images and their polar patches estimated by affine transform. (e) to (h): stereographic images and their polar patches estimated by affine transform.

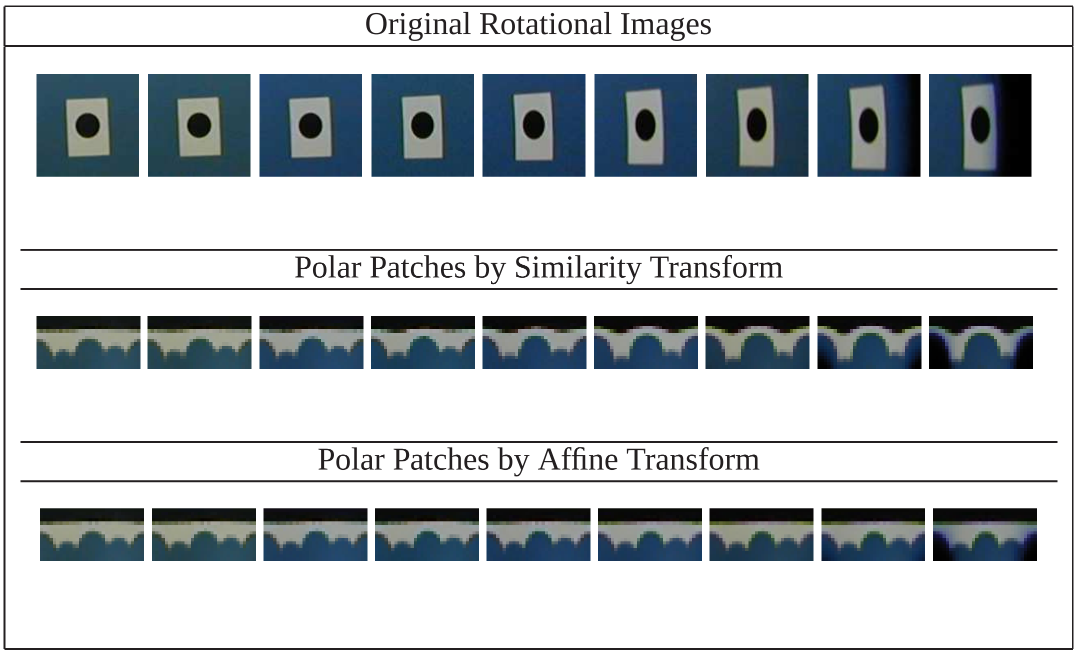


Figure 9: Original rotational images (top row), polar patches estimated by similarity transform (middle row), and polar patches estimated by affine transform (bottom row).

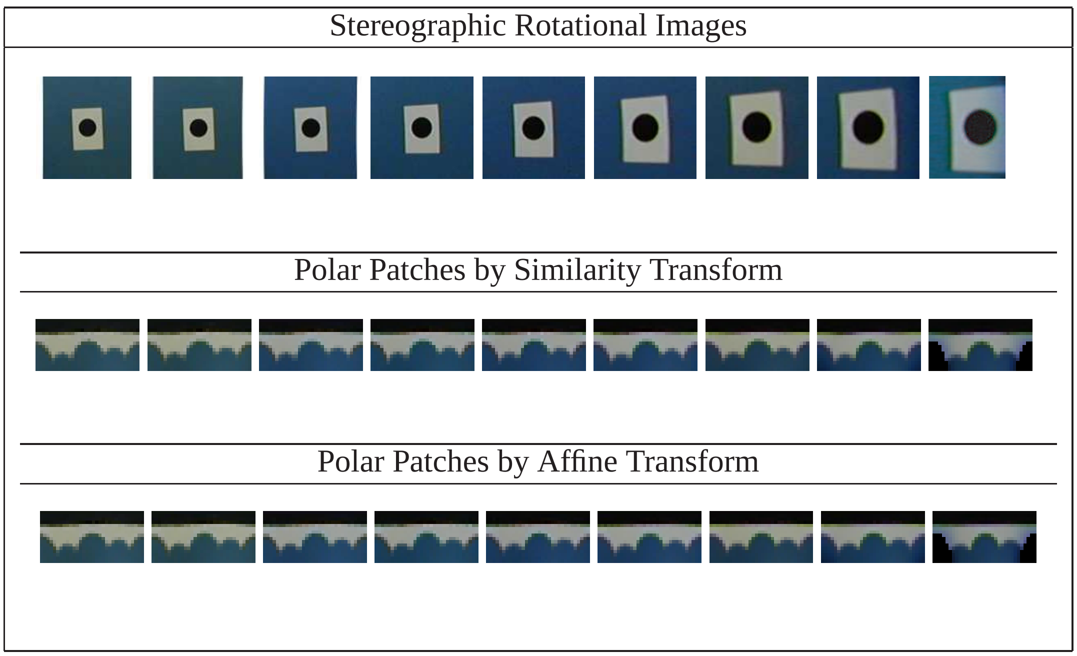


Figure 10: Stereographic rotational images (top row), polar patches estimated by similarity transform (middle row), and polar patches estimated by affine transform (bottom row).

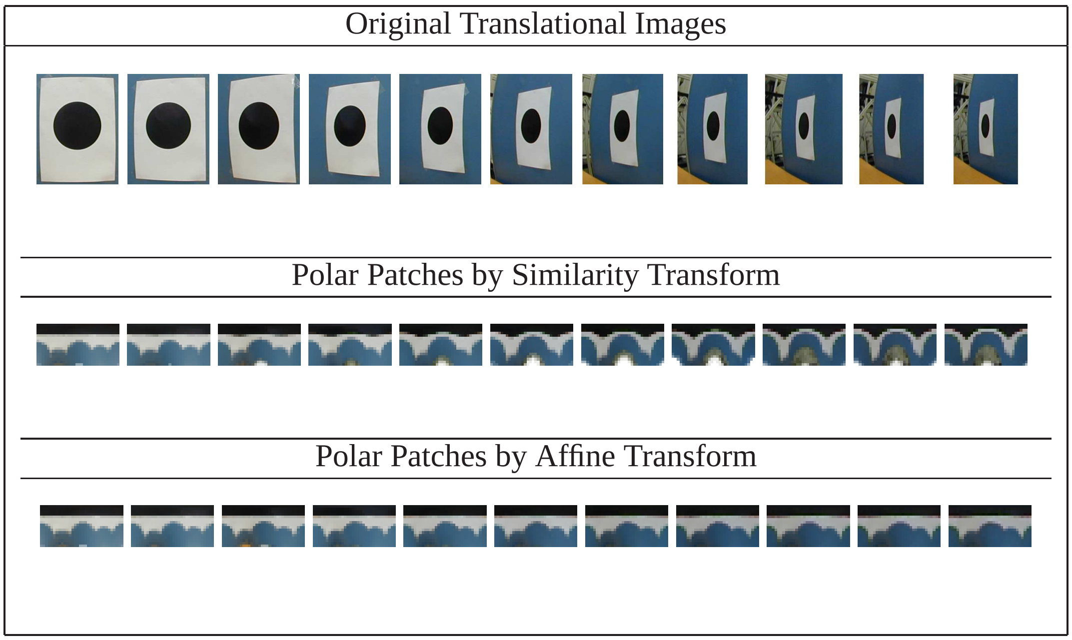


Figure 11: Original translational images (top row), polar patches estimated by similarity transform (middle row), and polar patches estimated by affine transform (bottom row).

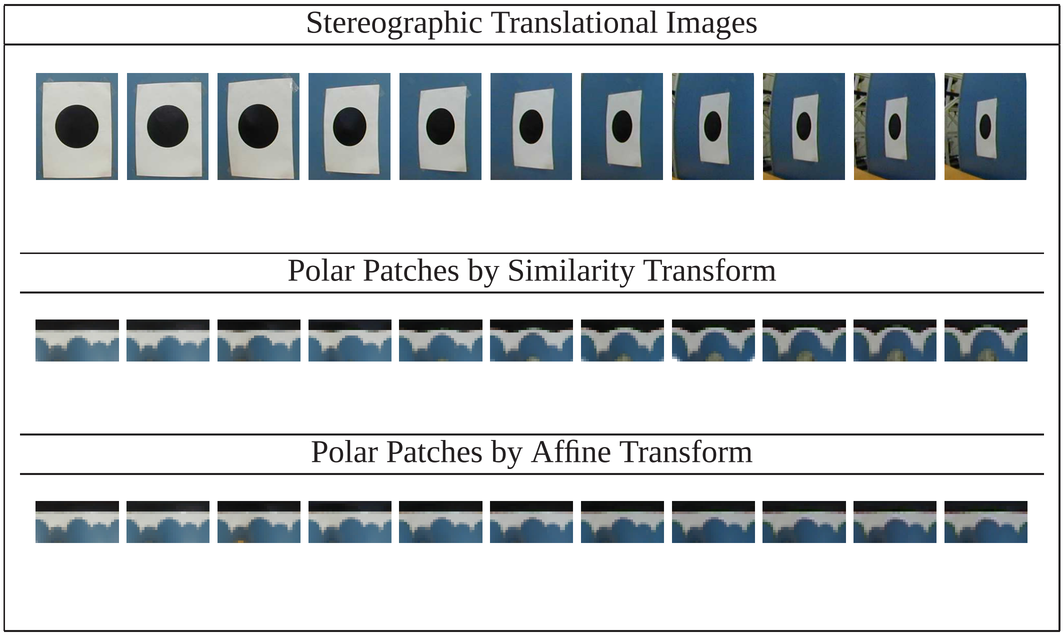


Figure 12: Stereographic translational images (top row), polar patches estimated by similarity transform (middle row), and polar patches estimated by affine transform (bottom row).

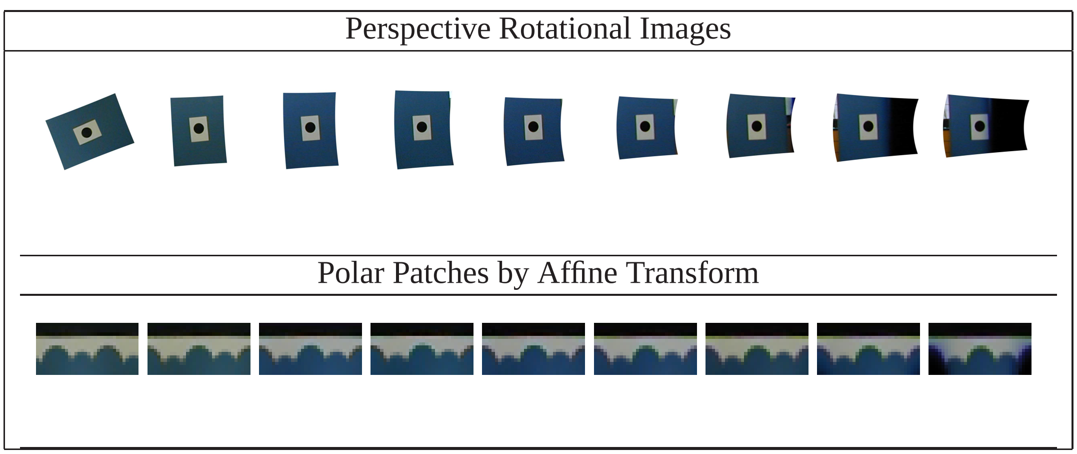


Figure 13: Perspective rotational images (top row), and polar patches estimated by affine transform (bottom row).

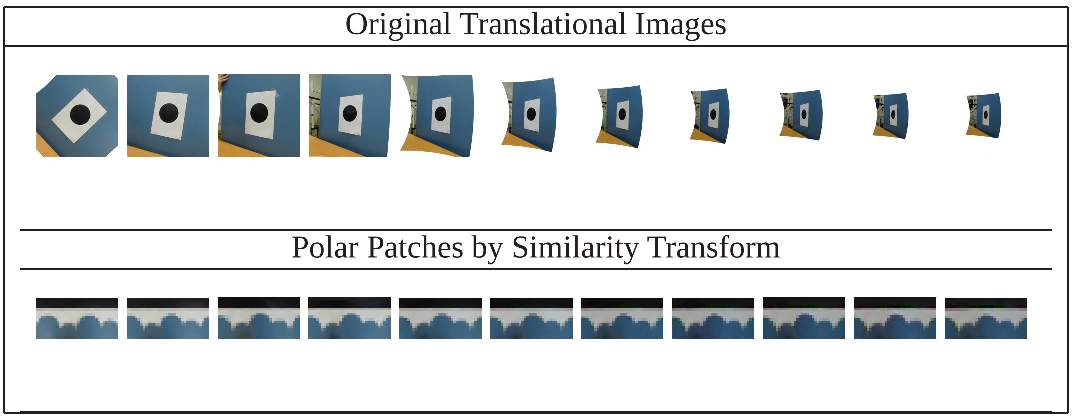
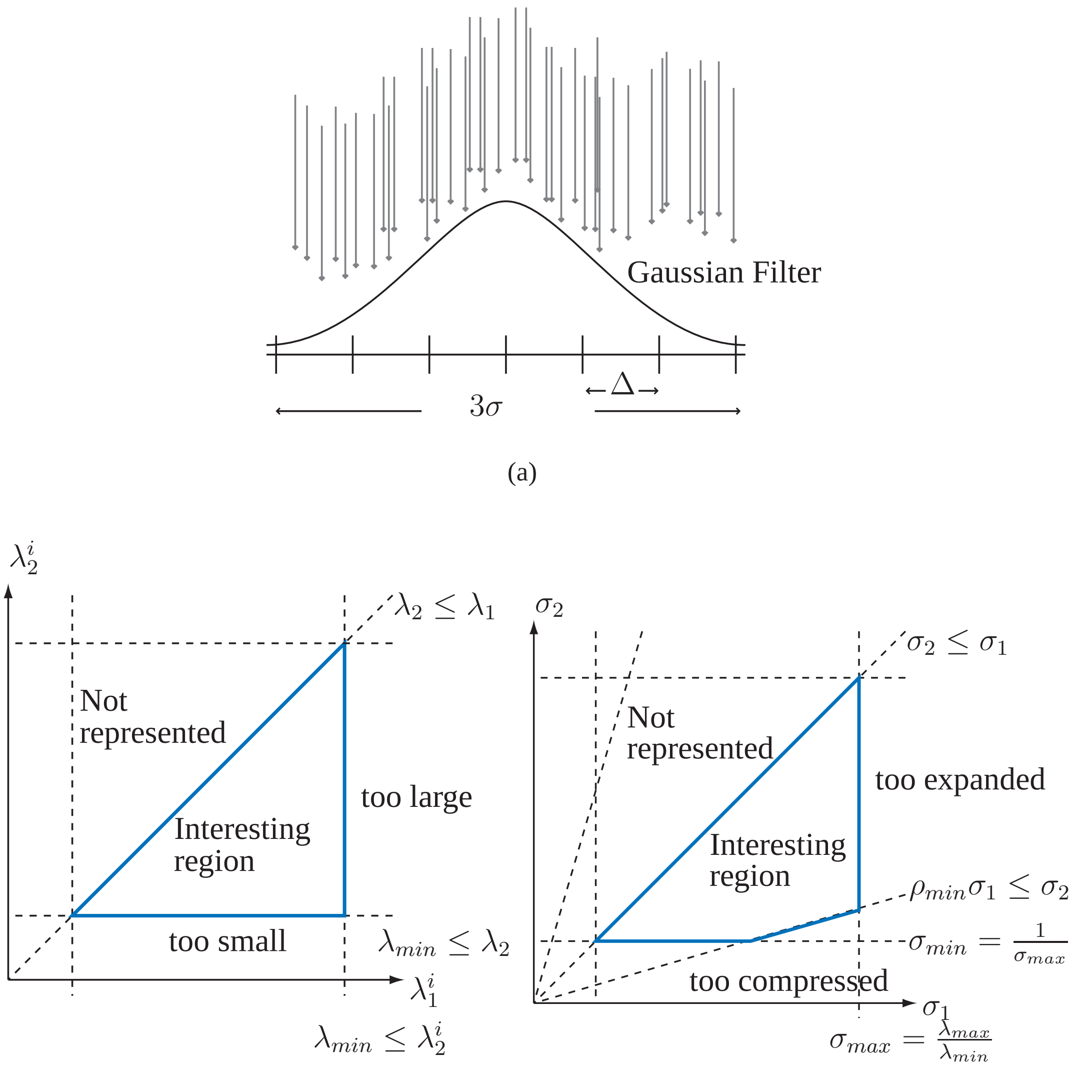


Figure 14: Perspective translational images (top row), and polar patches estimated by affine transform (bottom row).

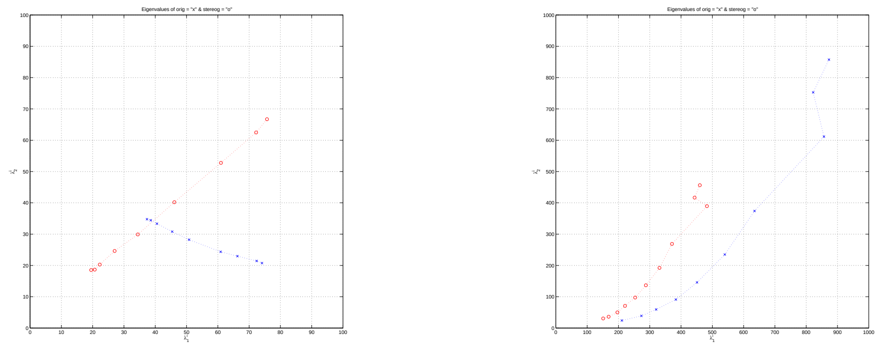


(a)

(b)

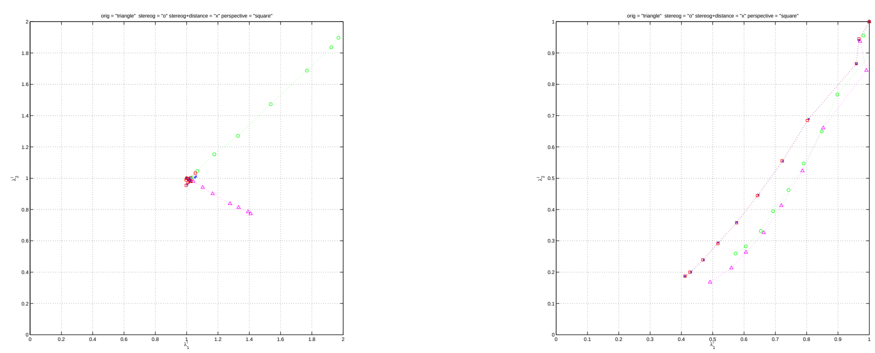
(c)

Figure 15: Interesting Regions.



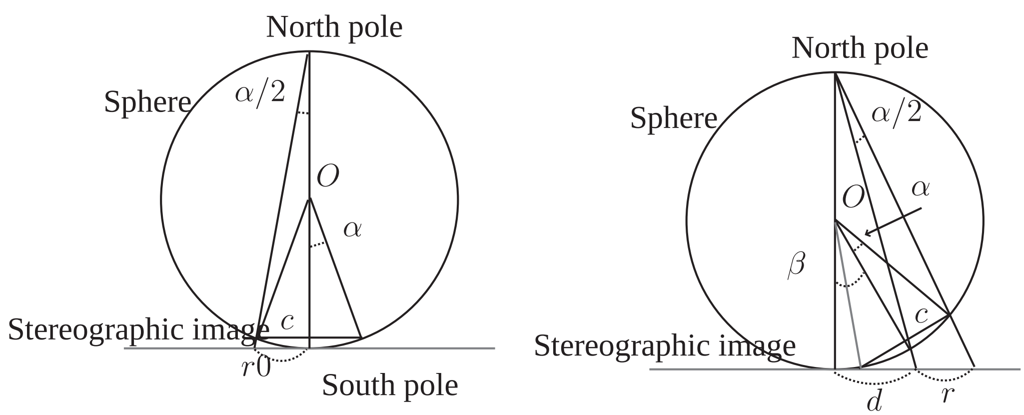
(a) (b)

Figure 16: λ of DR of the rotational (left) and translational (right) image sequence.



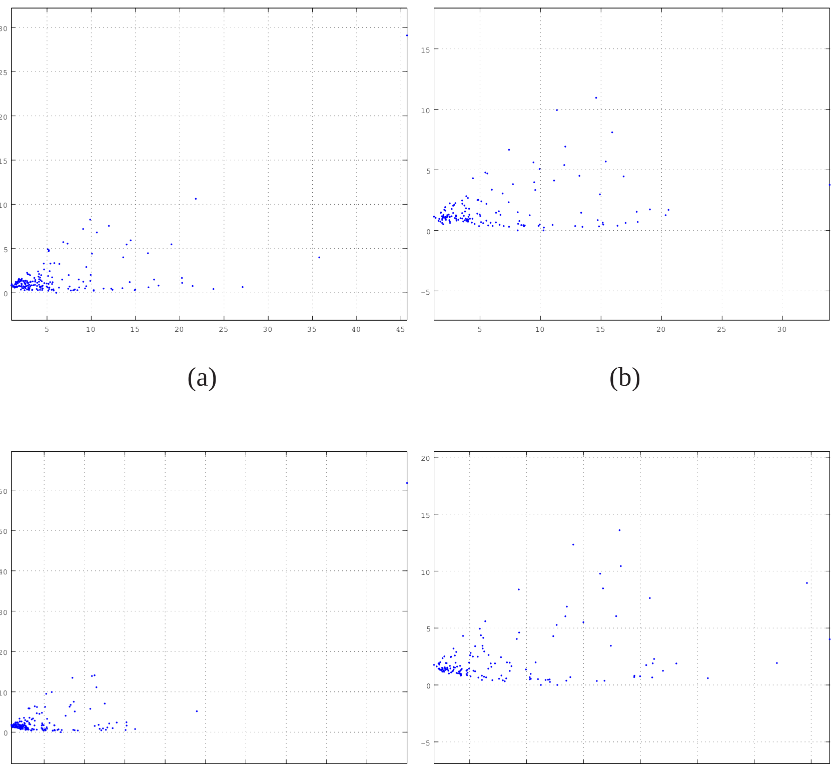
(a) (b)

Figure 17: σ of DR of the rotational (left) and translational (right) image sequence.



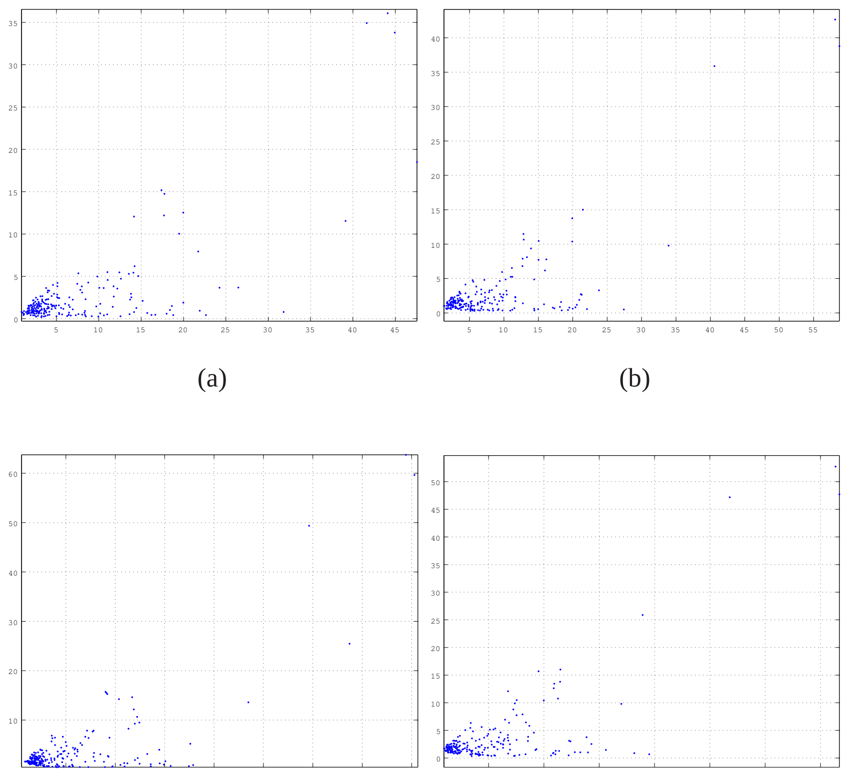
(a) (b)

Figure 18: Pure rotation and stereographic projection



(a) (b) (c) (d)

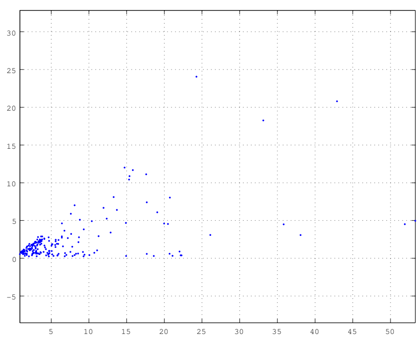
Figure 19: (a) and (b): σ_1 and σ_2 of all distinguished regions of the first original and stereographic image in Fig. 5. (c) and (d): λ_1 and λ_2 of all distinguished regions of the first original and stereographic image in Fig. 5.



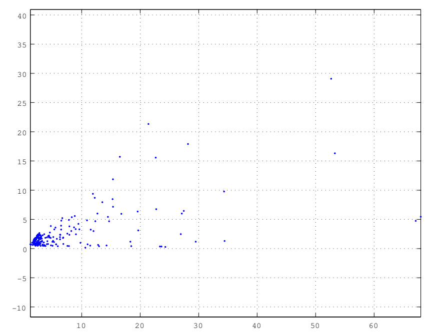
(a) (b)

(c) (d)

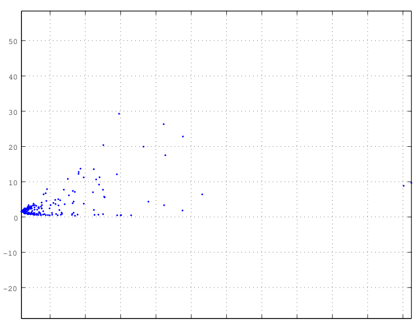
Figure 20: (a) and (b): σ_1 and σ_2 of all distinguished regions of the sixth original and stereographic image in Fig. 5. (c) and (d): λ_1 and λ_2 of all distinguished regions of the sixth original and stereographic image in Fig. 5.



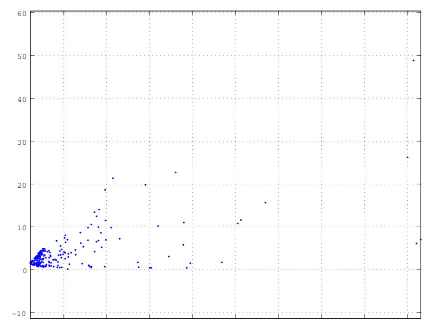
(a)



(b)

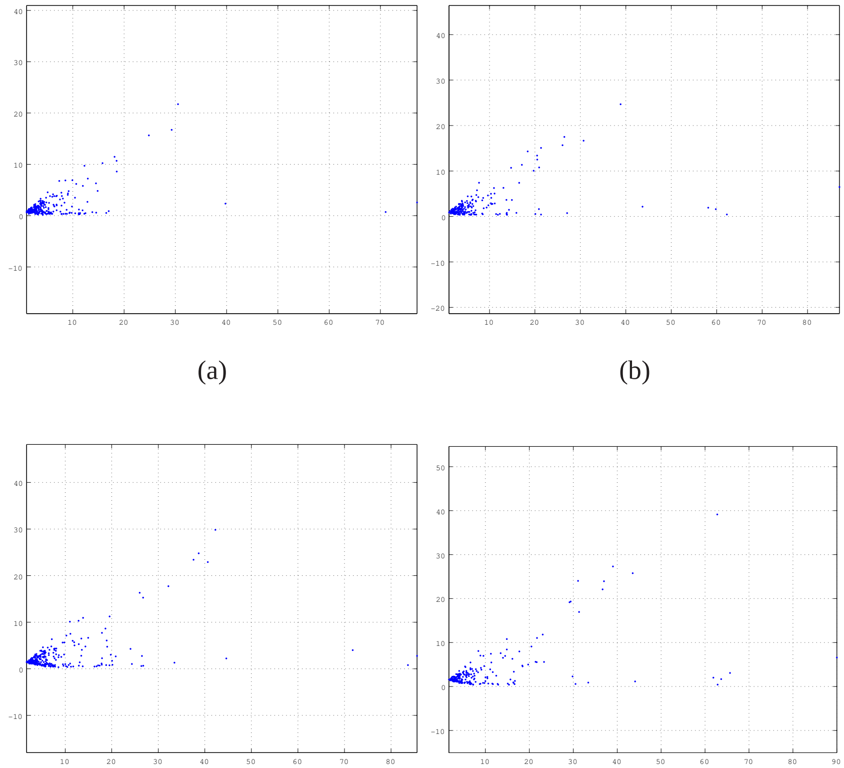


(c)



(d)

Figure 21: (a) and (b): σ_1 and σ_2 of all distinguished regions of the first original and stereographic image in Fig. 6. (c) and (d): λ_1 and λ_2 of all distinguished regions of the first original and stereographic image in Fig. 6.



(a) (b)

(c) (d)

Figure 22: (a) and (b): σ_1 and σ_2 of all distinguished regions of the eighth original and stereographic image in Fig. 6. (c) and (d): λ_1 and λ_2 of all distinguished regions of the eighth original and stereographic image in Fig. 6.

References

[1] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University, Cambridge, 2nd edition, 2003.

[2] Michael Henle. *Modern Geometries : Non-Euclidean, Projective, and Discrete*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2nd edition, 2001.

[3] A. Makadia and K. Daniilidis. Direct 3d-rotation estimation from spherical images via a generalized shift theorem. *IEEE CVPR*, June 2003.

[4] Jiří Matas, Petr Bílek, and Ondřej Chum. Rotational invariants for wide-baseline stereo. In H. Wildenauer and W. Kropatsch, editors, *Proceedings of the CVWW'02*, pages 296–305, Wien, Austria, February 2002. Pattern Recognition & Image Processing Group, Vienna University of Technology.

[5] Jiří Matas, Ondřej Chum, Martin Urban, and Tomáš Pajdla. Distinguished regions for wide-baseline stereo. Research Report CTU–CMP–2001–33, Center for Machine Perception, K333 FEE Czech Technical University, Prague, Czech Republic, November 2001.

[6] Jiří Matas, Ondřej Chum, Martin Urban, and Tomáš Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, September 2004.

[7] Branislav Mičušík and Tomáš Pajdla. Para-catadioptric camera auto-calibration from epipolar geometry. In Ki-Sang Hong and Zhengyou Zhang, editors, *Proc. of the Asian Conference on Computer Vision (ACCV)*, volume 2, pages 748–753, Seoul, Korea South, January 2004. Asian Federation of Computer Vision Societies.

[8] S.K. Nayar. Catadioptric omnidirectional cameras. In *CVPR97*, pages 482–488, 1997.

[9] Tomáš Svoboda, Tomáš Pajdla, and Václav Hlaváč. Epipolar geometry for panoramic cameras. In *Czech Technical University Workshop 98*, pages 177–178, Prague, Czech Republic, February 1998. Czech Technical University.