CENTER FOR
MACHINE PERCEPTION

CZECH TECHNICAL
UNIVERSITY IN PRAGUE

RESEARCH REPORT

ISSN 1213-2365

Towards Robot Localization and Obstacle Avoidance from Nao Camera

Michal Havlena, Šimon Fojtů, Daniel Průša,
Tomáš Pajdla

{havlem1, fojtusim, prusapa1, pajdla}@cmp.felk.cvut.cz

CTU–CMP–2010–18

November 5, 2010

Available at
ftp://cmp.felk.cvut.cz/pub/cmp/articles/havlena/Havlena-TR-2010-18.pdf

The work was supported by the EC project FP7-ICT-247525 HUMAVIPS. 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. 18, 2010

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

Towards Robot Localization and Obstacle Avoidance from Nao Camera

Michal Havlena, Šimon Fojtů, Daniel Průša, Tomáš Pajdla

November 5, 2010

Abstract

Nao is equipped with a bunch of sensors which help it to navigate inside a room without hitting the obstacles, namely the two cameras. We show the performance of the state-of-the-art structure from motion methods on the image sequences acquired by Nao's cameras, identify the major issues arising from the specifics of the data, and propose viable workarounds and method extensions in order to provide for future visual guidance of the robot. The actual robot control application is described in the appendix of the report.

1 Motivation

During Nao-to-human social interaction, the ultimate goal of HUMAVIPS project, Nao needs to move inside a room facing expected and unexpected obstacles while solving the assigned tasks. The computer vision methods that we want to deliver to Nao will make its movement easier by providing the following functionalities: (i) ability to reach the target even if it gets occluded for a while, (ii) ability to return to "home" position, and (iii) visual obstacle detection and avoidance.

Our aim is to use a state-of-the-art structure from motion method [13] (SfM) both for camera pose estimation and sparse 3D point cloud reconstruction. Having camera poses and knowing the rotation of the head, Nao's position and orientation can be computed using the transformation between the head and the torso of the robot. The obtained 3D point cloud may be used for visual obstacle detection because the position of the groundplane can be determined having the estimated robot trajectory.

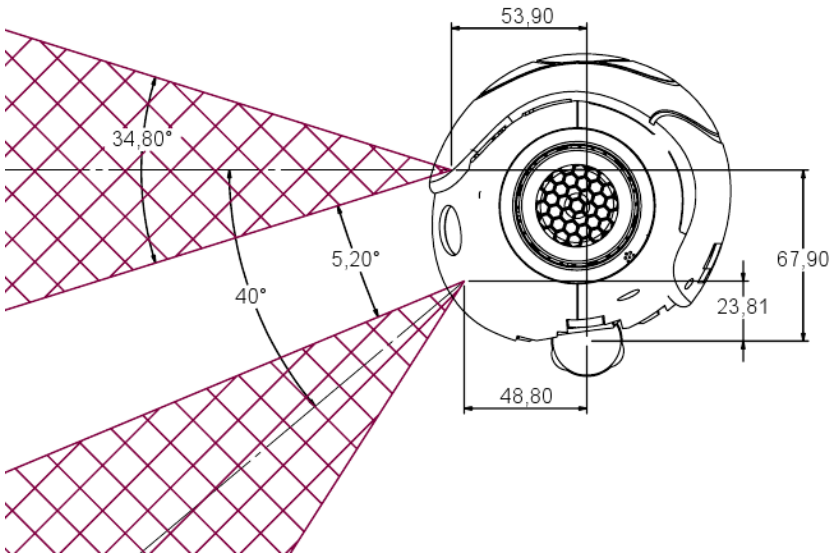


Figure 1: The configuration of the two CMOS cameras located on Nao's head does not allow stereo vision as their viewfields do not overlap. (Courtesy of [14]).

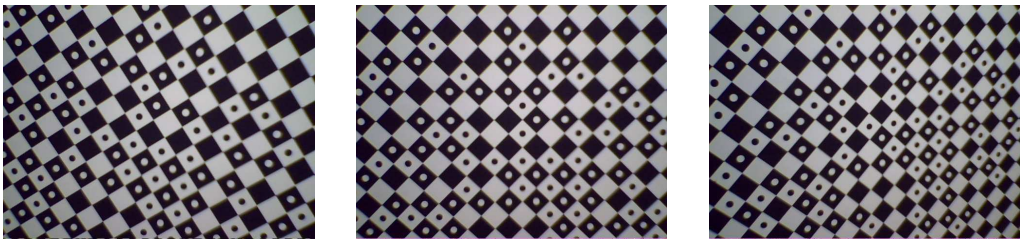
2 Towards SfM from Nao Camera

Next, we shall describe the specifics of Nao's cameras, the procedure of their calibration, and the first SfM experiments.

2.1 Nao camera properties

Two CMOS cameras having VGA, i.e. 640×480, resolution, which can capture up to 30 images per second, are located on Nao's head, see Figure 1. The configuration of the cameras does not allow stereo vision as their viewfields do not overlap – the first camera is heading forward and the other one downward in order to see directly in front of Nao [14].

Full camera framerate is achievable for the lowest resolution only, it is possible to capture at most 3 images per second using VGA resolution. As camera sensors are CMOS and have no shutter, images taken while the robot is moving have very low quality because different rows of the image come from different time points. As such images are not usable for geometric scene reconstruction, we decided to take images only when the robot is static and perform the shooting in a stop-and-go fashion.



(a) (b) (c)

Figure 2: Three images of a known calibration grid which were used to calibrate Nao's top camera. Pixel correspondence can be easily obtained due to the specific layout of white and black dots.

Due to the fact that both camera lens and sensor are tiny, the resulting images are rather noisy in the indoor environment because the camera auto-gain feature boosts the sensitivity of the sensor. This can be partially avoided by providing strong artificial room lighting mimicking outdoor light conditions.

2.2 Camera calibration

Internal camera calibration, which facilitates the transformation from image pixel coordinates to unit direction vectors, was obtained off-line [8]. Three images of a known calibration grid, see Figure 2, were used to compute calibration matrix and two parameters of radial distortion according to the polynomial model of degree two for both Nao's cameras independently.

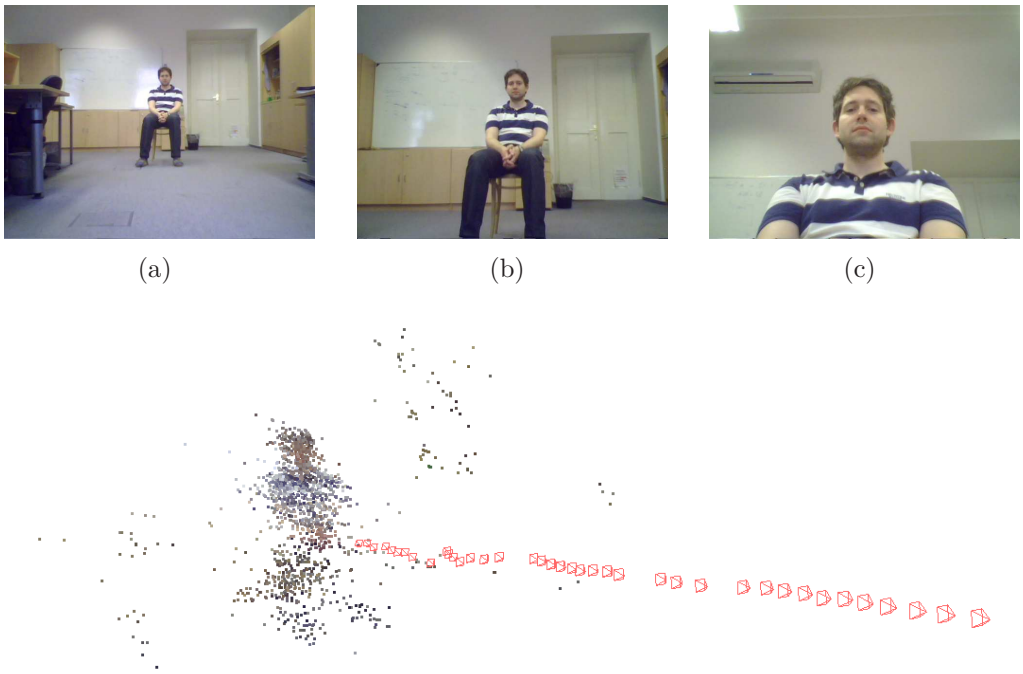
top camera:

$$K = \begin{bmatrix} 749.1048 & 0 & 329.3791 \\ 0 & 750.2743 & 227.1093 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

bottom camera:

$$K = \begin{bmatrix} 806.8948 & 0 & 282.0628 \\ 0 & 807.4827 & 223.9356 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

Images are radially undistorted before being used for further computation in order to improve SfM results.



(a) (b) (c)

(d)

Figure 3: Robot trajectory estimation from the top camera. (a)-(c) Sample images from the 39 images long sequence. (d) Reconstructed camera trajectory (red pyramids) and a sparse 3D point cloud (colored dots).

2.3 Robot trajectory estimation

We used a state-of-the-art sequential wide-baseline SfM method [13] to process a sequence consisting of 39 VGA images captured by Nao's top camera in a stop-and-go fashion while walking towards a static person sitting on a chair, see Figure 3(a)-(c). The total computation time was 11 minutes on a standard desktop PC which is roughly the time which was needed to capture the data itself, so the computation can be considered as being real-time. The estimated camera trajectory, see Figure 3(d), cannot be verified against ground-truth as no ground-truth measurement is available but it seems to be visually close to the true trajectory.

Our sequential SfM consists of several steps. First, MSER [9], SIFT [7], and SURF [1] salient image features are detected and described in input images. Tentative feature matches are obtained by the approximate nearest neighbour search in the descriptor space performed by FLANN [10] looking for the mutually best matches between pairs of consecutive images in the sequence. Next, pairwise matches are connected into tracks and only those

which form tracks longer than three images survive the filtering step. Robust pairwise epipolar geometries are computed by a voting scheme similar to the one used in [5] from ten runs of PROSAC [2] solving the 5-point minimal relative pose problem for calibrated cameras [11]. Finally, camera poses in a canonical coordinate system are recovered by chaining the epipolar geometries of pairs of consecutive images in the sequence [4] and a sparse bundle adjustment routine (SBA) [6] is applied to refine both the camera poses and the positions of the triangulated 3D points.

The major issue encountered in this experiment was the fact that most of the reconstructed 3D points were located on the person itself. This would ruin the camera pose estimation w.r.t the coordinate system of the room in a more realistic situation when the person was not completely static as the resulting camera poses would be estimated w.r.t the coordinate system connected with the person then.

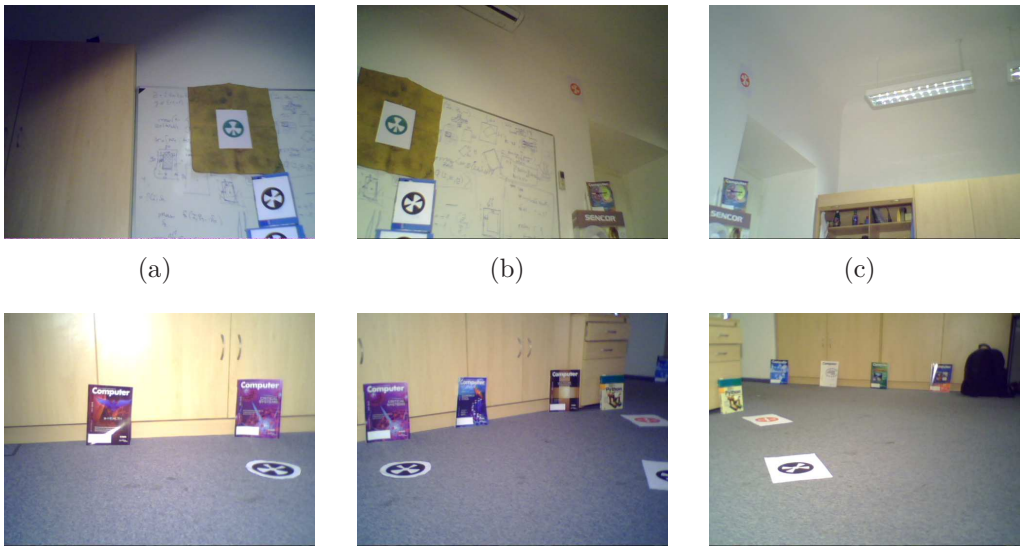
2.4 Extending camera field of view

We propose to extend the narrow field of view of the cameras by taking several images for different head rotations at each location where the robot stops as this should bring more image features originated in the room itself and overcome the aforementioned problem. We have implemented a routine which captures images by both Nao's cameras for three different head yaws ($33.5°$, $0°$, $-33.5°$) while the pitch of the head is set to $-30°$ in order to see the top and the bottom of the scene equally well, see Figure 4.

Geometric transformations between the individual images need to be known to be able to use the summation of the detected image features as the input to the structure from motion pipeline estimating the pose of a virtual camera looking straight forward. As the movement of the optical centers of the cameras during the rotation of the head is very small w.r.t the distance of the scene, we decided to model these geometric transformations as homographies, which assume fixed camera centers and pure rotation.

A homography between two images i and j is represented by a 3×3 matrix H_{ij} and can be computed by solving a linear system of equations built using four point correspondences between the given images [4]. Data normalization, RANSAC [3] searching for the model with the highest number of supporting tentative correspondences, and linear estimation from all the verified correspondences are used to increase the accuracy and robustness of the computation. It suffices to compute homographies between neighbouring image pairs because homography is a transitive operator:

$$H_{ik} = H_{jk} H_{ij}. \tag{1}$$



(a) (b) (c)

(d) (e) (f)

Figure 4: Six images captured using both Nao's cameras for three different head yaws at a fixed robot location. (a)-(c) Images taken by the top camera. (d)-(f) Images taken by the bottom camera.

Due to the fact that the images taken by Nao's cameras are of low quality and do not contain enough image features for reliable homography computation, we decided to find the parameters of the homographies off-line using a scene producing a sufficient number of features in the areas where the individual images overlap. Moreover, the images from the top camera have no overlap with the images from the bottom one, so three additional images located between the top and the bottom "row" of images have been acquired.

Homographies between six pairs of images have been computed and the parameters of the remaining transformations were obtained from the transitivity of homographies. The image created by stitching the six source images using the obtained transformations can be seen in Figure 5. The middle and bottom parts of the image were not contained in any of the source images and were impainted to improve the visual quality of the result.

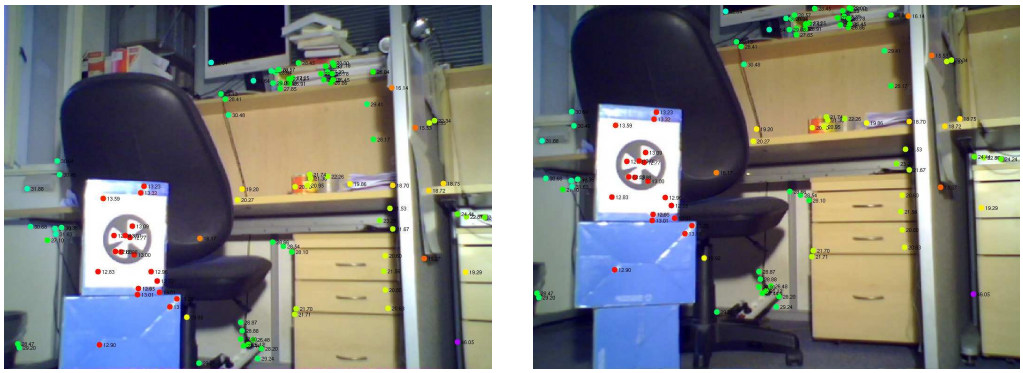
Unfortunately, first experiments with SfM from the summation of the detected image features have shown that the homographies obtained off-line do not always model the transformations between the individual images well. This could be caused by (i) an incorrect assumption that the movement of camera centers is negligible, or (ii) the fact that the head rotation is not always exactly the same, or (iii) the combination of both. We will investigate more in this direction in order to isolate and hopefully solve the problem.

Figure 5: The resulting wide field of view image stitched from the six narrow field of view images shown in Figure 4. The middle and bottom parts of the image were not contained in any of the source images and were impainted.

3 Depth Measurements from Nao Camera

When only the scene depth information for a single position of the robot is needed, it would be too computationally expensive and too slow to walk with the robot and to run the whole sequential structure from motion pipeline in order to retrieve the depths of the reconstructed 3D points. We can use a simpler SfM method, e.g. Bundler [12], to make a fast reconstruction from two images only in this case.

As the two Nao's cameras have no image overlap, we have implemented a routine which captures an image by the top camera when Nao is standing, lets Nao crouch, and takes another image by the top camera again. After inputting these images into Bundler, we get the resulting relative camera pose and a sparse 3D point cloud in 10 seconds. The reconstructed 3D points can be projected back to the images and their depths relative to the length of the baseline between the two camera centers can be evaluated, see Figure 6. Knowing the real length of the baseline, which is approximately 11 cm in this case, it is possible to compute the depths of the 3D points also in centimeters.



(a) (b)

Figure 6: Depth estimation from a pair of images. The color of each dot corresponds to the depth of the 3D point relative to the length of the baseline between the two camera centers (black numbers). (a) "Standing" image. (b) "Crouching" image.

4 Conclusion

We have performed the first step towards robot localization and obstacle avoidance from Nao camera. In our experiments, we have shown successful camera trajectory estimation from an image sequence captured by Nao's top camera and discussed the major issue encountered - the narrow field of view of the camera. We have proposed a method for extending the field of view via taking multiple images at each fixed robot location and summing the detected image features using homographies. Our future work will focus on improving structure from motion from the summed features and on the integration of the results of SfM into the robot control application which would provide for visual guidance of the robot.

We have also shown a fast yet reliable way of taking visual depth measurements from two images which could be extended into a detector of obstacles. This will be also a part of our future work.

References

[1] H. Bay, A. Ess, T. Tuytelaars, and L.J. Van Gool. Speeded-up robust features (SURF). *CVIU*, 110(3):346–359, June 2008.

[2] O. Chum and J. Matas. Matching with PROSAC: Progressive sample consensus. In *CVPR05*, pages I: 220–226, 2005.

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

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

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

[6] M.I.A. Lourakis and A.A. Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Tech. Report 340, Institute of Computer Science – FORTH, August 2004.

[7] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, November 2004.

[8] P. Mareček. A camera calibration system. Master's thesis, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, 2001.

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

[10] M. Muja and D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP09*, 2009.

[11] D. Nistér. An efficient solution to the five-point relative pose problem. *PAMI*, 26(6):756–770, June 2004.

[12] N. Snavely, S. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *IJCV*, 80(2):189–210, 2008.

[13] A. Torii, M. Havlena, and T. Pajdla. Omnidirectional image stabilization by computing camera trajectory. In *PSIVT 2009*, pages 71–82, 2009.

[14] Aldebaran Robotics – Nao Hardware Specification (Red Documentation Website). `http://academics.aldebaran-robotics.com/docs/site_en/reddoc/hardware/hardware.html`, 2010.

Appendix: Robot Control Application

The original purpose of the control application was to automatically guide the robot in an office-like environment and to scan the scene regularly. During implementation, we found out that the ultrasound sensors cannot be used to guide the robot reliably and it was decided to use manual guidance to control the robot. The application could be extended from a simple walk-and-scan to an application that allows a more complex control of the robot.

Controls

The robot can be controlled by keyboard and mouse in a CounterStrike®-like manner. The following list shows the keys and mouse moves used to control the robot. We give a full list, including some commands serving mainly for demo purposes, e.g. hail or talk.

- **arrows** go forward/backward, turn left/right
- **alt + left/right arrow** strafe left/right
- **shift + arrows** slow movement
- **mouse left-click** take screenshot from active camera
- **mouse right-click + movement** turning head
- **middle-click** align head to 0:0
- **s** stand up (blocking procedure; stable position)
- **c** crouch (blocking procedure; stable position)
- **h** hail with right arm
- **g** hail with left arm
- **r** release motors (all, except for head; use only in stable position!)
- **e** enslave motors (all, except for head)
- **w** release/enslave motors of head
- **i** toggle autorepaint of preview image
- **o** toggle bottom/top camera

- **p** toggle quality of preview image
- **a** acquire 6 screenshots (2 "rows" of 3 images; blocking procedure)
- **q** acquire 9 screenshots (3 "rows" of 3 images; blocking procedure; used for homography estimation)
- **t** talk (prompts for a text to say on console)
- **m** measures approximate distance to an object in image

Some of the procedures are blocking (as specified in the list), which means that the user cannot control the robot while such a procedure is being executed. The cause of this is that some of the procedures, e.g. acquiring images, need the robot not to move in order to work properly. Other procedures, e.g. stand up or crouch, need to be finished before the user can continue controlling the robot. On the other hand, some actions can be performed simultaneously, e.g. greeting with arm while walking. It must be kept in mind that the stability of the robot is greatly influenced by hand moves and it is up to the operator to prevent the robot from falling.

Description

The program, implemented in C++, is a remote application, i.e. not running on the robot. It is composed of three main parts. The first part handles the initialization of all important proxies. A while loop in the second part reads the inputs from keyboard and mouse and controls the robot appropriately. Third part of the program ensures a stable position while releasing motors and disconnects from the proxies. There is also a fail-safe mechanism to safely shut down the robot, when a **ctrl + c** is pressed.

Usage

This application allows a simple teleoperation of the robot with visual feedback. It has a functionality to acquire images, which are needed for robot trajectory estimation via SfM. This program is also useful for a quick and simple control of the robot. It is quite a common need that the operator needs the robot to perform a trivial operation, e.g. stand up or release motors. One way is to launch Choregraphe®, a multi-functional application which takes some time to start and connect to the robot. On the other hand, launching our control application is a matter of a few seconds and the robot is operable and ready to use.