Combining Central and Peripheral Vision for Reactive Robot Navigation*

Antonis A. Argyros
Computer Vision and Robotics Lab.
ICS/FORTH
Heraklion - Crete - Greece
argyros@ics.forth.gr

Fredrik Bergholm
Computational Vision and Active Perception Lab.
NADA/KTH
Stockholm - Sweden
fredrikb@nada.kth.se

Abstract

In this paper, we present a new method for vision-based, reactive robot navigation that enables a robot to move in the middle of the free space by exploiting both central and peripheral vision. The robot employs a forward-looking camera for central vision and two side-looking cameras for sensing the periphery of its visual field. The developed method combines the information acquired by this trinocular vision system and produces low-level motor commands that keep the robot in the middle of the free space. The approach follows the purposive vision paradigm in the sense that vision is not studied in isolation but in the context of the behaviors that the system is engaged as well as the environment and the robot's motor capabilities. It is demonstrated that by taking into account these issues, vision processing can be drastically simplified, still giving rise to quite complex behaviors. The proposed method does not make strict assumptions about the environment, requires very low level information to be extracted from the images, produces a robust robot behavior and is computationally efficient. Results obtained by both simulations and from a prototype on-line implementation demonstrate the effectiveness of the method.

1 Introduction

The term navigation refers to the capability of a system to move autonomously in its environment by using its own sensors. The more specific term visual navigation is used for the process of motion control based on the analysis of data gathered by visual sensors. The topic of visual navigation is of particular importance mainly because of the rich perceptual input provided by vision. Moreover, navigation that is based on other types of sensors, in contrast to vision, often requires modification of the environment (e.g. insertion of emitters) which imposes constraints on the application of such methods in unknown environments.

The problem of visual navigation has been traditionally treated without taking very much into account the environment of the robot, its body and the characteristics of the desired behavior. Typically, monocular or stereoscopic visual systems are assumed and the effort is focused on constructing a general representation of the environment that may thereafter support the solution of any vision-related problem. During the last decade, a new vision paradigm has attracted the interest of the computational vision research community. According to this paradigm, called active and purposive vision [1], vision is more readily understood in the context of the behaviors in which the system is engaged. Consequently, vision attemtps to explore the aspects of the world that are important for the system at a given point in time, instead of aiming at a general representation of the environment which, besides being extremely difficult to extract, it is probably not needed either. The interest in purposive vision is largely motivated by the fact that all biological vision systems are highly active and purposive [2]. The purposiveness of visual processes enables the formulation and the solution of simpler problems that have a relative small number of possible solutions and can be treated in a qualitative manner [3].

In this paper, we describe a new method for visual robot navigation based on the principles of purposive vision. By employing a forward-looking camera for central vision and two side-looking cameras for sensing the periphery of the visual field, reactive robot navigation has been achieved. The developed method combines the information acquired by this trinocular vision system to produce low-level motor commands that keep the robot in the middle of the free

*This research has been carried out during the first author's 1996-97 appointment to CVAP/NADA/KTH and was funded under the VIRGO research network (EC Contract No ERBFMRX-CT96-0049) of the TMR Programme.

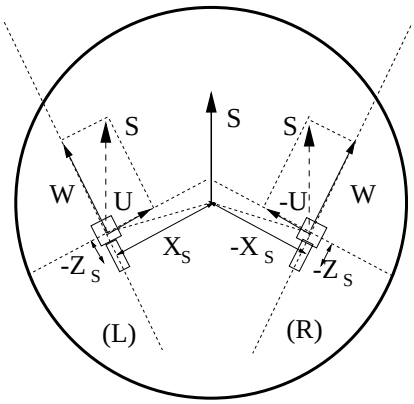

Figure 1. Top-down view of the robot geometry. The placement of the two peripheral cameras is also shown.

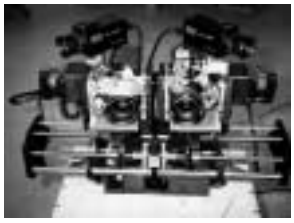

Figure 2. The KTH head with two extra cameras mounted on it for implementing peripheral vision.

space. The aim of this work is to investigate how the design of a visual system can assist robots with specific bodies and motor capabilities in exhibiting particular behaviors. It is demonstrated that considering the behavior and the motor capabilities of a robot when designing its visual system, leads to theoretically simpler and computationally more efficient solutions.

The rest of the paper is organized as follows. Section 2 describes the requirements that the target behavior poses to the design of the robot's visual system. Section 3 presents issues related to the motion information that can be computed by each camera as well as how this information is processed and used to drive the robot. Section 4 presents results from simulations of the method as well as implementation issues and results obtained by an on-line implementation on a real robotic platform. Finally, in section 5, the main conclusions of this work are summarized and future research plans are described.

2 The behavior, the environment and the body

The study of the behaviors that should be exhibited by an observer, its environment as well as the specifics of the observer, provides valuable hints on how the sensors should be placed in order to facilitate the implementation of a particular behavior. In this work we assume a robot that can translate in the forward direction and rotate (pan) around its vertical axis (Fig. 1). We aim at developing a vision based reactive navigation capability that enables the robot to navigate in flat-floor indoor environments (long corridors, narrow passages, rooms), avoiding collisions with walls and obstacles. The term reactive is used to express lack of a particular destination that could be set by using maps of the environment, landmark recognition etc. Free space is defined based on the motor capabilities of the robot: the robot moves on a plane and, therefore, all 3D structures that do not belong to this plane can be potentially harmful if the robot crashes on them. Since the robot is about to "live" in indoor environments, it is expected to be able to handle situations where long corridors and narrow passages are encountered. It can be shown that difficulties arise when only central vision is used (i.e. a camera or a fixating stereo configuration at the direction of translation). Consider for example a camera with a field of view of 30 degrees that is placed in a a 2-meters wide corridor with its optical axis parallel to the walls. The camera can only see walls that are approximately 4 meters ahead and it is therefore quite difficult to maneuver accurately. On the other hand, the use of cameras with wide field of view [4] give rise to depth dependent geometric distortions that are difficult to correct. In order to implement this behavior, it appears quite natural to exploit the information provided by peripheral vision, i.e. visual information at large angles with respect to the direction of forward translational motion (see for example the configuration in Fig. 2). By using such a camera configuration, the robot is able to perceive walls and obstacles that are immediately close to it. Moreover, the target behavior may be implemented by indirectly comparing crude structure information acquired by the left and right peripheral cameras instead of computing precise structure information. This approach is motivated by experiments that study the behavior of honeybees [5]. In these experiments, bees were trained to navigate along corridors towards a source of food. The bees were observed to navigate in the middle of the corridor. The eyes of the bees are pointing laterally (at about 180 degrees). The behavior is based [5] on velocity information computed at the left and right eyes of the bee. In simple terms, if a non-rotating (no panning/tilting) bee is in the center of the corridor, it perceives the world as "leaving" its optical field with the same velocity in both eyes, while if the bee is closer to one of the sides of the corridor, it perceives it as moving faster. For a non-rotating observer, the difference in the observed velocities depends only on depth.

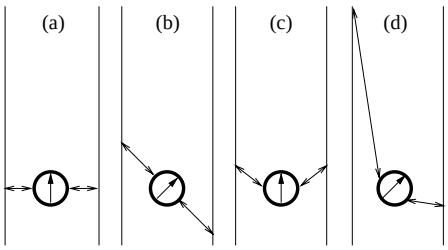

Figure 3. (a), (b) If a robot with lateral peripheral cameras is in the middle of the free space, it perceives equal distances from left and right walls independently of pose. (c) and (d) If the peripheral cameras are slanted, the distances at left and right are equal only if the robot's pose is parallel to the walls.

Therefore, balancing the flow, balances the distances to the left and the right of the observer. Santos-Victor et al. [6] proposed the divergent stereo approach in order to exploit this finding in robots. They exploit visual information that is captured by two cameras with optical axes of opposite orientation that are mounted perpendicularly to the direction of forward translation. Our research differs to the approach in [6] in several ways. First, peripheral cameras are not placed in opposite directions because decisions on forward motion should not be influenced by "past" structure information. Proximity calculations based on data at 90 degrees angle to the motion direction are sort of obsolete for most reactive schemes and navigation situations. Second, it turns out that control is facilitated when the cameras are slanted. See for example Figs. 3(a) and 3(b) where the cameras are placed laterally on the robot body. The robot perceives equal distances at its left and right side, independently of pose. However, the situation is different in cases 3(c) and 3(d) where the cameras are slanted towards the direction of translation. If the robot is in the middle of the free space, it perceives equal distances from the walls only if its pose is parallel to the walls. Therefore, in this case, flow balancing fixes also the pose of the robot. Last, but not least, we study the effects of the observer's rotational motion in the flow computed by the two peripheral cameras. A moving robot is not only translating but also rotating and this rotation affects the computed flow. We also show how central vision (i.e. visual information acquired in the direction of the translation) can be used along with peripheral vision in order to simplify the problems to be solved.

3 Method description

Consider an arbitrary 3D reference coordinate system (RCS). Consider also a 3D camera coordinate system (CCS) that is positioned at the optical center (nodal point) of a pinhole camera. Assume that the center of the RCS remains fixed at coordinates (X_s, Y_s, Z_s) with respect to the CCS. If the RCS moves with 3D translational velocity (U, V, W) and 3D rotational velocity (α, β, γ), the equations relating the 2D velocity (u, v) of an image point $p(x, y)$ to the 3D motion parameters of the projected 3D point $P(X, Y, Z)$ are [7]:

$$\begin{aligned} u &= \frac{-Uf + xW - \alpha x Y_s + \beta(xX_s + Z_s f) + \gamma Y_s f}{Z} \\ &+ \alpha\frac{xy}{f} - \beta\left(\frac{x^2}{f} + f\right) + \gamma y \end{aligned} \tag{1}$$

$$\begin{aligned} v &= \frac{-Vf + yW - \alpha(yY_s + Z_s f) + \beta y X_s + \gamma X_s f}{Z} \\ &+ \alpha\left(\frac{y^2}{f} + f\right) - \beta\frac{xy}{f} - \gamma x \end{aligned}$$

where f denotes the focal length of the camera.

The projection of the optical flow (u, v) along the intensity gradient direction (i.e. the perpendicular to the edge at that point) is also known as normal flow. The normal flow is less informative than optical flow but can be computed robustly and efficiently from image sequences by just using differentiation techniques. Moreover, in contrast to the computation of optical flow, no environmental assumptions such as smoothness are required for normal flow computation. For the above reasons, the proposed method for reactive robot navigation relies on the computation of the normal flow field. Let (n_x, n_y) be the unit vector in the gradient direction. The magnitude u^M of the normal flow vector is given by:

$$u^M = n_x u + n_y v \tag{2}$$

By substituting Eqs. (1) in Eq. (2) we obtain:

$$\begin{aligned} u^M &= -n_x f\left(\frac{U + \beta Z_s + \gamma Y_s}{Z} - \beta\right) \\ &- n_y f\left(\frac{V - \alpha Z_s + \gamma X_s}{Z} + \alpha\right) \\ &+ (xn_x + yn_y)\left(\frac{W + \beta X_s - \alpha Y_s}{Z} - \frac{x}{f}\beta + \frac{y}{f}\alpha\right) \\ &+ (yn_x - xn_y)\gamma \end{aligned} \tag{3}$$

In our robot setup (see Fig 1), we set the RCS on the robot's body so that the Y axis coincides with the robot's rotational axis and the Z axis is parallel to the robot's translational motion. As it has already been discussed, we assume that the robot is capable of translating with velocity S and rotate (pan) with velocity β. We set the nodal point of the right peripheral camera so that the center of the RCS is at coordinates $(-X_s, 0, -Z_s)$. Similarly we set the nodal points

of the left peripheral camera and of the central camera so that the center of the RCS is at coordinates $(X_s, 0, -Z_s)$ and $(0, 0, 0)$, respectively. The translational velocity S of the robot produces translational velocities $(-U, 0, W)$, $(U, 0, W)$ and $(0, 0, S)$ for the right, left and central cameras, respectively. For all three cameras, the rotational velocity of the robot produces a rotational velocity $(0, \beta, 0)$ at the camera coordinate system.

By taking into account the above considerations for each of the left (L), right (R), and central (C) cameras, Eq. (3) gives:

$$\begin{aligned} {u_L}^M &= \left(\frac{-U-\beta Z_s}{Z_L} - \beta\right)(-n_x f) \\ &+ \left(\frac{W+\beta X_s}{Z_L} - \beta\frac{x}{f}\right)(xn_x + yn_y) \quad (4) \\ {u_R}^M &= \left(\frac{U-\beta Z_s}{Z_L} - \beta\right)(-n_x f) \\ &+ \left(\frac{W-\beta X_s}{Z_L} - \beta\frac{x}{f}\right)(xn_x + yn_y) \quad (5) \\ {u_C}^M &= \beta n_x f \\ &+ \left(\frac{S}{Z_C} - \beta\frac{x}{f}\right)(xn_x + yn_y) \quad (6) \end{aligned}$$

where Z_L, Z_R and Z_C represent the depth of the 3D points perceived by the left, right and central cameras, respectively. By selecting normal flow vectors for which it holds that $xn_x + yn_y = 0$[1], we obtain:

$$f_{LD} = \frac{U-\beta Z_s}{Z_L} - \beta \quad , \quad f_{RD} = \frac{-U-\beta Z_s}{Z_R} - \beta \quad (7)$$

and

$$f_{CD} = \beta \quad (8)$$

and and by selecting normal flow vectors for which it holds that $(n_x, n_y) = (0, 1)$ (the vertical normal flows), we obtain:

$$f_{LH} = \frac{W+\beta X_s}{Z_L} - \frac{\beta x}{f}, f_{RH} = \frac{W-\beta X_s}{Z_R} - \frac{\beta x}{f} \quad (9)$$

The left sides of Eqs. (7), (8) and (9) can be computed from images because they employ normal flow values, point coordinates and gradient directions. The right sides of Eqs. (7) and (9) employ functions of depth and can get an even simpler form by noting that the function acquired by the central camera (Eq. (8)), gives the rotation. Thus, Eqs. (7) and (9) after derotation, become:

$${f^d}_{LD} = \frac{U-\beta Z_s}{Z_L} \quad , \quad {f^d}_{RD} = \frac{-U-\beta Z_s}{Z_R} \quad (10)$$

[1]The selected normal flow vectors are those that are tangent to circles centered at the image origin

and

$${f^d}_{LH} = \frac{W+\beta X_s}{Z_L} \quad , \quad {f^d}_{RH} = \frac{W-\beta X_s}{Z_R} \quad (11)$$

Thus, central vision can be used to derotate the flow fields produced at the peripheral cameras. Having exploited this observation, it turns out that:

$$({f^d}_{LD} + {f^d}_{RD})\frac{X_s}{Z_s} - ({f^d}_{LH} - {f^d}_{RH}) = \left(U\frac{X_s}{Z_s} - W\right)\left(\frac{1}{Z_L} - \frac{1}{Z_R}\right) \quad (12)$$

Equation (12) can be rewritten in a simpler form as follows:

$$\mathcal{F} = \mathcal{C}\left(\frac{1}{Z_L} - \frac{1}{Z_R}\right) \quad (13)$$

In Eq. (13), $\mathcal{F}$ is a quantity that can be directly computed from functions of normal flow that have been extracted from the central and peripheral cameras. $\mathcal{C}$ is an unknown constant (of known sign) that depends on the characteristics of the body of the observer as well as its constant translational velocity. Function $\mathcal{F}$ is equal to zero when the left and right cameras are in equal distances from world points and takes positive or negative values depending on whether the right camera is farther or closer from obstacles compared to the left camera. Therefore, the computable quantity $\mathcal{F}$ can be used to control the rotational velocity by keeping the quantity $\mathcal{F}$ as close to zero as possible, achieving this way the desired behavior. Note that $\mathcal{C}$ is equal to zero if the nodal points of the left, right and central cameras are collinear because in this case $U\frac{X_s}{Z_s} = W$. In our robot setup we avoid this special case.

4 Implementation issues - Experimental results

An experimental evaluation of the proposed method has been based on both simulation results as well as on results obtained by an on-line implementation of the method on a real robotic platform. Simulations have been based on the KHEPERA simulator [8], which has been modified to simulate the central and peripheral cameras of the robot. The aim of the simulation experiments was to test the control law used to drive the robot. Thus the function $\mathcal{F}$ of Eq. (13) has been simulated and the robot was set to navigate in various environments. Several experiments were conducted. Figure 4 shows a sample run. Thin dark lines represent the walls of the corridor-like environment. The thick dark line is the trace of the robot. It can be observed that the robot started at the bottom-right end of the environment and, after reaching the end of the corridor has started moving

Figure 4. A run of the simulated robot.

backwards. Moreover, the robot moves in a smooth path among the various obstacles of the environment.

One of the most interesting results of simulations was the difference in the behavior of the simulated robot depending on whether the peripheral cameras were laterally placed or slanted with respect to the direction of translation. It turns out that slanted cameras result in smooth robot paths, while the laterally placed ones produce snake-like robot motion patterns.

The simulation experiments are, of course, not adequate for testing the performance of the method when real vision processes are employed. For this reason, an on-line implementation of the method has been realized. The platform used was a LABMATE ROBUTER on which "Charlie", the KTH active vision head has been mounted. Two extra cameras were mounted on "Charlie" implementing peripheral vision (Fig. 2). Only one of the central cameras was used to implement central vision. Note that the central camera is not placed at coordinates $(0, 0, 0)$ with respect to the RCS as it has been theoretically assumed. The process of selecting normal flow vectors at specific directions leads to cancelling of motion components. It turns out that Eq. (8) still holds if $X_s \neq 0$.

In our implementation, a SUN Ultra Sparc was responsible for peripheral vision processing and a PENTIUM processor running LINUX was responsible for central vision processing. The distributed processing as well as the interprocess communication was based on the TCX communications library [9]. Various navigation scenarios have been tested in which the robot successfully managed to perform maneuvers in narrow passages. In Figs. 5 and 6, we present snapshots from two different navigation sessions.

In our present algorithm, we did not allow for individual motions of the cameras (eye movements). This is roughly tantamount to the assumption of approximately known FOE while exhibiting this behavior. However, we did not calibrate the head so that the central camera pointed exactly in the forward motion translation direction. In fact, we noticed that in many of the successful navigation experiments the optical axis of the central camera was 5–10 degrees off the

Figure 5. Snapshots of a navigation session (left to right, top to bottom).

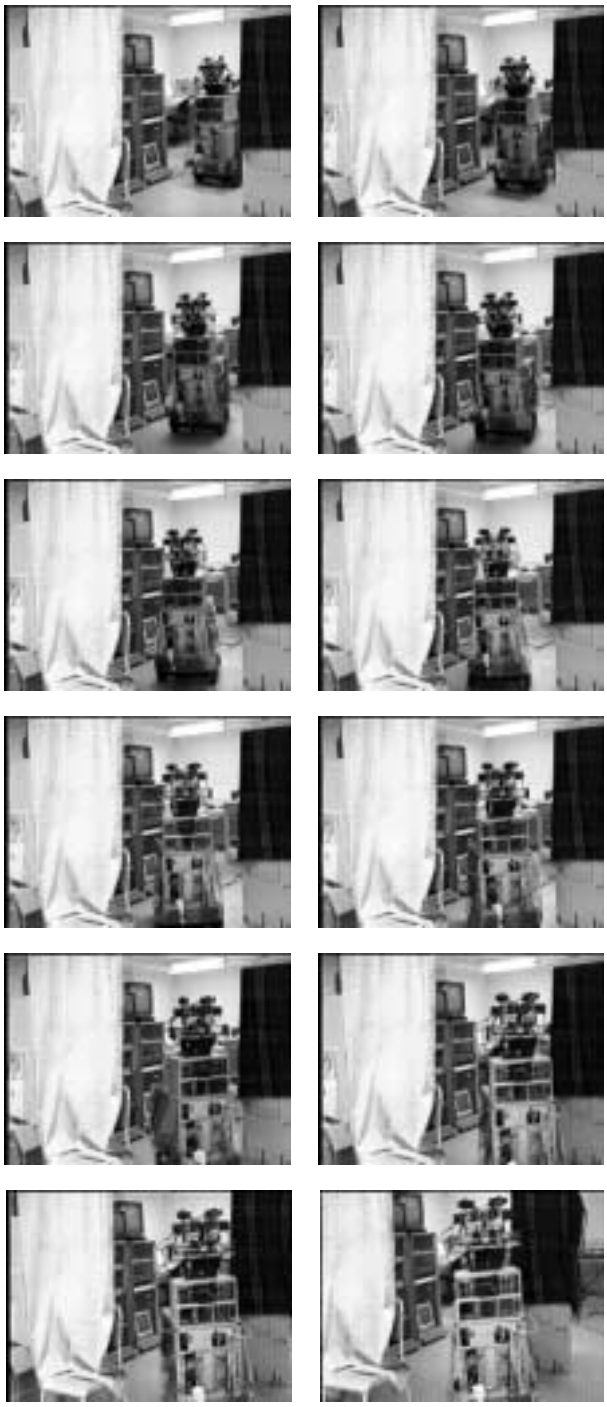

Figure 6. Snapshots of a navigation session (left to right, top to bottom).

forward motion direction.

5 Conclusions

A method has been proposed that enables a robot to navigate in free space based on a combination of central and peripheral vision. The method does not make strict assumptions about the environment, it requires very low level information to be extracted from the images, it produces a robust robot behavior and it is computationally efficient. Results obtained by both simulations and from a prototype on-line implementation demonstrate the effectiveness of the method. Peripheral vision seems to be very useful for achieving certain behaviors and its combination with central vision seems natural and appears to be powerful. Future research work will investigate ideas on further exploiting combinations of central and peripheral vision.

References

[1] Y. Aloimonos, I. Weiss and A. Bandopadhay, "Active Vision", *IJCV*, Vol. 1(4), pp. 333-356, 1988.

[2] R.A. Brooks, "Intelligence Without Reason", *AILAB Memo 1293*, MIT AI Lab, April, 1991.

[3] Y. Aloimonos, "Purposive and Qualitative Active Vision", *Proceedings of DARPA Image Understanding Workshop*, pp. 816-828, 1990.

[4] T. Camus, D. Coombs, M. Herman and TH Hong, "Real-Time Single-Workstation Obstacle Avoidance Using Only Wide-Field of View Divergence", *ICPR*, 1996.

[5] M.V. Srinivasan, M. Lehrer, W.H. Kirchner and S.W. Zhang, "Range Perception through Apparent Image Speed in Freely Flying Honeybees", *Visual Neuroscience*, Vol. 6, pp. 519-535, 1991.

[6] J Santos-Victor, G. Sandini, F. Curotto and S. Garibaldi, "Divergent Stereo in Autonomous Navigation: From Bees to Robots", *IJCV*, Vol. 14(2), pp. 159-177, March 1995.

[7] H.C. Longuet-Higgins and K. Prazdny, "The Interpretation of a Moving Retinal Image", *Proceedings of the Royal Society, London B*, pp. 385-397, 1980.

[8] O. Michel, *"Khepera Simulator 2.0 User Manual"*, University of Nice, Sophia Antipolis.

[9] C. Fedor, *"TCX: An Interprocess Communication System for Building Robotic Architectures"*, Users Manual, 1994.