A neural field approach for robot motion control

Conference Paper · November 1998

DOI: 10.1109/ICSMC.1998.726556 · Source: IEEE Xplore

A Neural Field Approach to Robot Motion Control

Percy Dahm[1], Carsten Bruckhoff[2], and Frank Joublin[3]
Institut für Neuroinformatik, Ruhr-Universität Bochum
44780 Bochum, Germany
Tel: +49 234 700 [1]7969 / [2]5570 / [3]7974
Fax: +49 234 709 4209
Email: [1]Percy.Dahm / [2]Carsten.Bruckhoff / [3]Frank.Joublin@neuroinformatik.ruhr-uni-bochum.de

Abstract— **In this article we introduce a biologically inspired approach to robot motion control. It is based on a so-called neural field which can be described by a nonlinear competitive dynamical system. Movement directions are assigned to the field's artificial neurons by employing codebook vectors. Due to the field's intrinsic dynamical properties the problem of reaching a goal under constraints can be solved efficiently.**
Our approach is validated by applications to local navigation and manipulator control.

Keywords— **robot motion planning, control, neural field, nonlinear dynamics, local navigation, manipulation, time to contact**

I. Introduction

The basic task in robot motion planning is to endow a robot with the ability to reach a goal under constraints, e.g. moving towards a target while avoiding obstacles. Approaches that have been made can coarsly be devided into global and local methods. Global methods are typically graph or grid based. They return a continuous free path considering the entire environmental information.

By contrast, local methods need only local information. The complete path results from iterating the robot's current state. Due to their low computational costs and their ability to react quickly to environmental changes, they are much more suitable for practical robotic applications. The most popular local method is the so-called potential field approach pioneered by Kathib [9]. The idea is to represent the robot's state as a point in configuration space moving under the influence of an artificial potential field. The robot's goal is generated by an attractive potential, and constraints are realized by repulsiv potentials. The force field which is applied to the robot is obtained by calculating the negated gradient of the total potential. The major drawback of potential field methods is that spurious local minima can arise. These are points where the individual potentials cancel.

Schöner et.al. [12] proposed a local navigation approach based on dynamical systems. They represented movement directions of a mobile robot by a one dimensional potential field. The robot's heading direction is a behavioral variable which is controlled by a single dynamical equation. To avoid the problem of local minima arising from their obstacle representation they introduced additional nonlinear dynamics.

In the course of this paper we present a new approach to robot motion control. Like the potential field approach it is a local method. Since it runs while the robot is moving, we prefer the notion of control instead of planning. In contrast to potential field methods a movement step is not calculated by following the negated gradient, but through a neural mechanism which actively selects a movement direction from a set of possible directions. It is interesting to mention that our basic idea can be partially found in the biological motor system: Recording the electrical activity of single neurons from the parietal and frontal cortices of monkeys, Georgopoulos et. al. [6] found a correlation between neural activity and the direction of the movement of the arm.

The neural mechanism we apply is based on the so-called neural field introduced by Amari [1]. It is described by an integrodifferential equation which can be discretized to obtain a nonlinear competitive dynamical system affecting a set of artificial neurons. Although Amari's original intention was to model cortical neurophysiology, several authors have rediscovered neural fields for the purpose of robotics: Engels [5], [12] applied them to memory representation in local navigation. Furthermore, for controlling a mobile robot's heading direction he used an approximation which reduces the field dynamics to a single dynamical equation. Bicho [2] used a neural field to stabilize a mobile robot's behavior regarding phototaxis. Finally, Bruckhoff and Dahm [3] developed a model for autonomous decision making in local navigation, including aspects of active information acquisition and processing.

In the following, we give a more detailed description of neural fields. Based on simulations we will present applications to local navigation and manipulation.

II. What is a neural field?

A neural field can be described by the following dynamical equation:

$$\tau \dot{u}(\varphi) = -u(\varphi) - h + s(\varphi) + \int_{-\infty}^{\infty} w(\varphi, \varphi')\sigma(u(\varphi'))d\varphi' \quad (1)$$

Equation (1) stands for the onedimensional limit case of a continuous field of artificial neurons. A neuron is characterized by an activation $u(\varphi)$ and an input $s(\varphi)$, called stimulus. The interaction between neurons located at φ and φ' is given by the integral term in eq.(1). $w(\varphi, \varphi')$

denotes the interaction kernel. Typically[1], it is chosen proximally excitatory and distally inhibitory, e.g., a mexican hat function.

The interaction kernel is followed by a step function[2] $\sigma(u(\varphi'))$:

$$\sigma(u(\varphi')) = \begin{cases} 1 & : \quad u(\varphi') \geq 0 \\ 0 & : \quad u(\varphi') < 0 \end{cases} , \tag{2}$$

From this follows that only neurons that have an activation greater equal zero can contribute to the integral term. Considering only the linear part in eq.(1), $u(\varphi)$ converges to $s(\varphi) - h$. Hence h fixes the threshold of $\sigma(u(\varphi'))$. Finally, τ determines the time scale of the dynamic.

A neuron activation depends on its stimulus $s(\varphi)$ and on the activation of other neurons lying within the range of interaction. Wether they increase or decrease the regarded neurons activation is given by $w(\varphi, \varphi')$. Obviously, eq.(1) can lead to clusters of activation, called localized solutions. Amari has shown that his equation has - among others - three basic types of solutions: homogenous solutions (Φ-solutions, the whole field is activated or deactivated), an instable localized solution (a1-solution), and a stable localized solution (a2-solution). The latter one keeps stable, even if the stimulus is removed. Moreover, it is capable of following the stimulus field's negated gradient. For mathematical details see [1], [10]).

III. Application to local navigation

For numerical reasons eq.(1) has to be discretised:

$$\tau \dot{u}_i = -u_i - h + s_i + \sum_{j=0}^{N} w_{ij} \sigma(u_j) \tag{3}$$

where N is the total number of neurons, and the indices denote movement directions.

We use the neural field to control the robot's planar movements regarding target acquisition and obstacle avoidance. Hence, it has to encode angles from $-\pi$ to π. By means of a codebook we represent a set of discrete directions by twodimensional unit vectors $\hat{r}_i$, $i = 1 \cdots N$. Note, that we use a coordinate system which has one of its axis fixed to a global reference direction (see fig.1). If we fixed it to the robot's heading direction, we would represent angular velocities, not movement directions.

In order to calculate the interaction kernel w_{ij}, we have to determine the angular distance δ_{ij} between two vectors $\hat{r}_i$,$\hat{r}_j$:

$$\delta_{ij} = \arccos(\hat{r}_i \hat{r}_j) \tag{4}$$

Since the field is periodic, the interaction kernel has to be periodic, too:

$$w_{ij} = k \cdot e^{-\delta_{ij}^2/\gamma^2} - H_0 \tag{5}$$

[1] It is possible to apply more complicated kernels, see [7], for example.

[2] For making interaction smoother it is recommended to replace the step function by a sigmoidal function

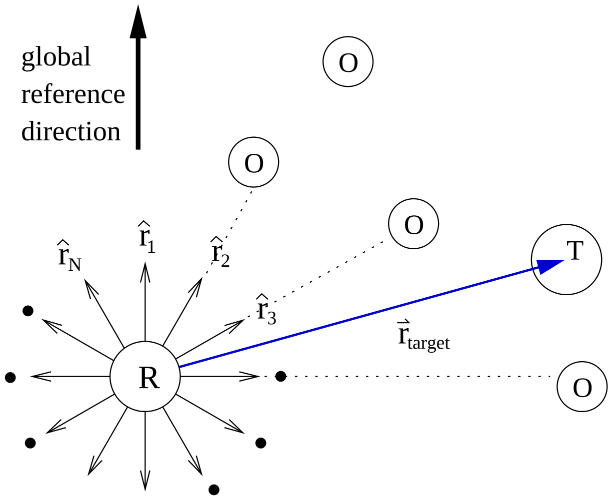

Fig. 1. R=robot, T=target, O=obstacles

The range of excitation is fixed by γ, its amplitude by k. Naturally, the robot cannot move in more than one direction. Therefore, we have to choose a global inhibition H_0, which allows only one localized peak in the field.

In order to let the field select an appropriate direction of movement, the field's stimulus has to be provided with the necessary information:

First, we set s_i to a value s'_i which reflects the angular distance between the target vector $\vec{r}_{target}$ (see fig.1) and the corresponding movement direction $\hat{r}_i$ ($C' = const$):

$$s'_i(\vec{r}_{target}) = \frac{C'}{2} \left(\frac{\vec{r}_{target} \hat{r}_i}{|\vec{r}_{target}|} + 1 \right) \tag{6}$$

Second, obstacles are considered if their distance to the robot (d_i) is less than a threshold d_{thres} which is discussed below. In this case s_i is set to a constant negativ value s''_i ($s''_i = -const$). Filling the stimulus field can be done algorithmically or, at once, by using the following function:

$$s_i(\vec{r}_{target}, d_i) = s'_i(\vec{r}_{target}) \cdot (1 - \theta_{d_i < d_{thres}}) + s''_i \cdot \theta_{d_i < d_{thres}}, \tag{7}$$

where $\theta_{d_i < d_{thres}}$ is a step function:

$$\theta_{d_i < d_{range}} = \begin{cases} 1 & : \quad d_i < d_{thres} \\ 0 & : \quad d_i \geq d_{thres} \end{cases} , \tag{8}$$

If no obstacle exists with respect to $\hat{r}_i$, we assume d_i to be infinite. The range in which obstacles are considered is set to the robot's current distance to the target ($d_{thres} = \vec{r}_{target}$). If we would choose a constant range, for example, the robot could never approach targets, which have an obstacle lying behind them regarding the robot's target direction.

In practical applications d_{thres} is additionally limited by the available sensor range.

Finally, the most activated neuron subscripted by w

$$w = argmax(u_i | i \in [1, N]) \tag{9}$$

encodes the movement direction which is executed by the robot. Typically, a mobile robot is controlled by its translatory (v_T) and rotatory velocity (v_R). While the latter one is easy to obtain, we have to discuss the translatory velocity: In obstacle free situations it can be choosen constant ($v' = const$). However, close to obstacles the robot needs to be slowed down. This can be easily achieved by taking the field's maximum activation into account:

$$v_T = v'\sigma(u_w), \tag{10}$$

where $\sigma(u_w)$ is a sigmoidal function:

$$\sigma(u_w) = \frac{1}{e^{-\beta u_w} + 1} \tag{11}$$

The sigmoids slope β can be experimentally adjusted. Applying eq.(10) leads to a reasonable overall behavior: As shown below the field's activation reflects some kind of confidence regarding the selected movement direction, hence, the translatory velocity is adapted to the environmental situation.

Finally, it is to note that in all our considerations, here and below, control takes place in configuration space, i.e. the robot is described by a point given by its current configuration. Followingly, obstacles and other constraints have to be mapped into configuration space, too. For most mobile robots this can be done by increasing the angular interval obstacles cover from the robots point of view. In the below discussed manipulator control, we introduce a more sophisticated description of constraints in configuration space.

Simulation of local navigation

Our neural field approach to motion control has been successfully implemented on our robot 'ARNOLD'[3] (see fig. 6). Its only sensor is an active vision system. Without moving, this sensor covers only a limited angle interval. Hence, we introduced gaze control and a low level short term memory (see [3], for details).

Since these additions are quite hardware specific, we prefer to present simulation results to discuss general aspects of our approach:
In order to update the entire stimulus field at each time step we assume a 360 degree sensor. Fig.2 shows the path of a robot moving from initial to target position. The robot's maximum sensor range is depicted by a circle. Its radius is 50 arbitrary units.

Figure 3 and 4 illustrate the temporal course of activation and stimulus, respectively. Due to the range limitations of the sensor the robot initially moves straight towards the target (time step 0-90). Since no obstacles are detected, both, the stimulus field as well as the activation field are unimodal (maximum depicted in white, minimum in black). In the following time steps the stimulus contains obstacle entries. Therefore, it becomes bimodal. By

[3]The anthropomorphic autonomous mobile robot ARNOLD is used as a testbed in the NEUROS project supported under grant 01IN504F from the German Ministry of Education and Research.

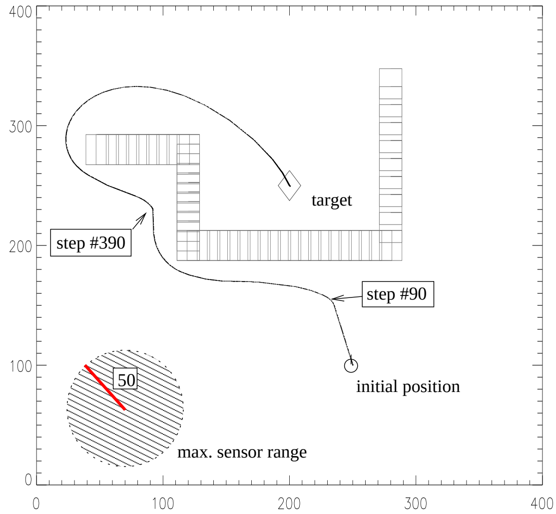

Fig. 2. Robot's arena

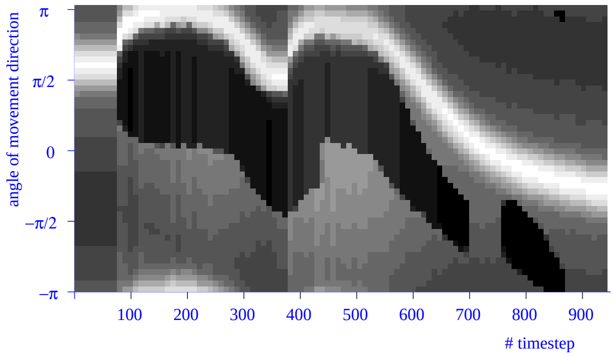

Fig. 3. Temporal course of activation

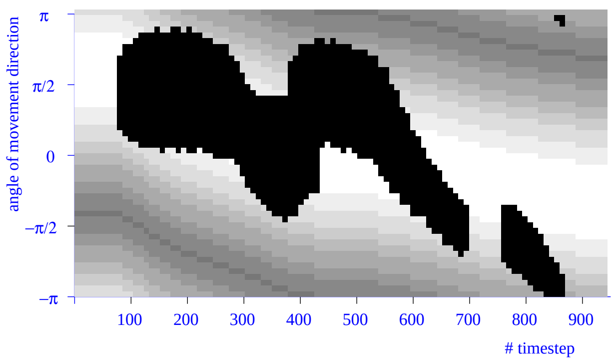

Fig. 4. Temporal course of stimulus

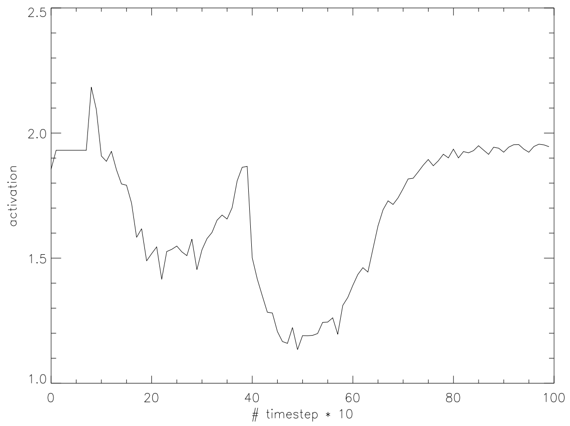

Fig. 5. Max. activation

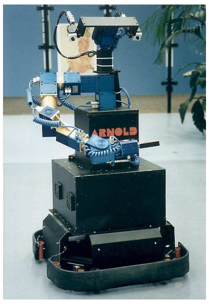

Fig. 6. ARNOLD

contrast, the activation field stays unimodal. This crucial fact is based on competition within the neural field. Furthermore, the field's parameters are adjusted such that the localized peak is of a_2-type (see section II), i.e. it is stable despite a small stimulus (regarding the peak) and a much greater local stimulus maximum voting for almost the opposite direction. The peak follows smoothly the local optimum given by target acquisition and obstacle avoidance.

Figure 5 shows the activation field's maximum in its temporal course. Obviously, it decreases if the peak is driven out of target direction by negative obstacle stimuli (see time step 90 and 390). Moreover, imagine a robot being entirely surrounded by obstacles: in this case the peak collapses, no neuron has an activation greater than zero. Therefore, the field is capable of representing that no appropriate direction can be selected. According to eq.(10) the robot stops until the environmental situation changes.

IV. Application to manipulation

Motion planning for robot manipulators can be done in cartesian or in joint space. In both cases a desired task state (endeffector position and orientation) has to be reached. If we consider the biological motor system to be the optimal solution to effector control problems, it is useful to study human arm movements: Experiments in psychophysics have shown that a goal directed movement seems to be controlled in cartesian space. Without perturbation, the hand's tip always moves on a straight line between initial and final position [8]. If control would take

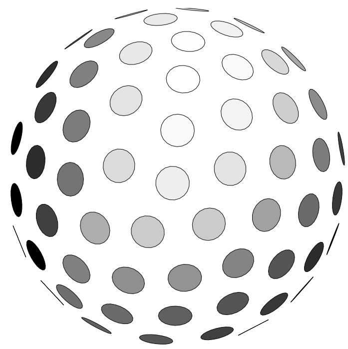

Fig. 7. Homogeneous neuron distribution on a sphere's surface

in joint space, the shortest path between initial an final joint configuration would lead to a curved trajectory in cartesian space.

Hence, all following considerations refer to cartesian space.

Our robot 'ARNOLD' is equipped with a 7 degree of freedom (dof) anthropomorphic manipulator. The basic task of endeffector position and orientation control is augmented by positioning the elbow. Due to kinematical constraints it is forced to move on a circle around shoulder wrist axis. Its position on this circle can be described by an angle α, which is called arm angle. We have developed a closed form solution for the inverse kinematics [4], which directly maps the task specification consisting of endeffector position (3dof), its orientation (3dof), and the arm angle (1dof) to joint coordinates (7dof).

In order to simplify the control problem we keep the above mentioned task decomposition. First, we focus on the problem of position control. The scheme decribed in sec. III can be almost identically applied. While in the planar (2D) problem the codebook vector's endpoints lie on a circle around the robot's position, in 3D, they lie on a sphere's surface. To obtain a homogeneous representaion of this surface, we applied a neural gas algorithm [11] to learning a set of random movement directions with 114 artificial neurons. Our result can be seen in fig.(7). Each circle represents an artificial neuron. (The sphere's opposite side is kept hidden, for clarity). Figuratively speaking, this sphere is centered at the endeffector's tip. Like in section III, its orientation is fixed to a global coordinate system.

To get the target related stimulus part we use eq.(6). Since we do not control a single point moving in 3D, but the endeffector's position, obstacle avoidance has to be extended regarding the entire manipulator. Using a geometrical model allows to calculate the minimum distance between obstacles and manipulator. We do not consider obstacles which have a distance to the manipulator greater than the minimum one. This restriction is useful to reduce computational costs.

Our approach needs obstacle information with respect to each direction $\hat{r}_i$, hence having only the manipulator's current distance to obstacles is not sufficient. A well suited way to solve this problem is to use the quantity 'time-to-contact'. It represents the time in which the manipulator will have contact with an obstacle, if the endeffector's tip

moves in a specified direction with a constant velocity. In the following we will give a linear approximation to obtain the time-to-contact t_i^c for a direction $\hat{r_i}$:
Let $d(\vec{p_0}, t)$ be the minimum distance regarding an endeffectors position p_0 at time t. A virtually executed movement into direction r_i leads to a new position $\vec{p_i}(t + \delta t)$:

$$\vec{p_i}(t+\delta t) = \vec{p_0}(t) + v\hat{r_i}\delta t, \tag{12}$$

where v denotes the translatory velocity. The minimum distance as a function of time can be linearly approximated:

$$d(t) \simeq d(\vec{p_0}(t)) + \frac{d(\vec{p_i}(t+\delta t)) - d(\vec{p_0}(t))}{\delta t} t \tag{13}$$

For $d(t) = 0$, which corresponds to contact, eq. 13 yields:

$$t_i^c = \frac{d(\vec{p_0}(t))\delta t}{d(\vec{p_0}(t) - d(\vec{p_i}(t+\delta t))} \tag{14}$$

If the minmum distance increases ($d(\vec{p_0}(t)) > d(\vec{p_i}(t+\delta t))$), t_i^c becomes negative, i.e. the corresponding direction is obstacle free. Hence, we set t_i^c equal to infinity.
As discribed in section III, we consider the target's distance. The corresponding time-to-contact t^c_{target} is:

$$t^c_{target} = \frac{|\vec{r}_{target}|}{v} \tag{15}$$

According to equation(7) the stimulus field can be filled as follows:

$$s_i(\vec{r}_{target}, t_i^c) = s_i'(\vec{r}_{target}) \cdot (1 - \theta_{t_i^c < t^c_{target}}) + s_i'' \cdot \theta_{t_i^c < t^c_{target}} \tag{16}$$

The same method we described here for the endeffector's position has to be done for its orientation and the arm angle. Only the topology of the corresponding fields changes. If we neglect the control of the gripper orientation (which is only of vanishing use for obstacle avoidance) we obtain a two dimensional control problem. The latter one can be solved by using the same topology as the one applied to local navigation. Finally, the arm angle control is a one dimensional problem. Since there are only two possible movement directions, we need only two neurons. Applying our approach to this simple problem carries almost exclusively the advantage of methodical consistency. Apart from the benefit of a temporal filter, here, selecting the stimulus maximum works as well.

Regarding general manipulator control further constraints have to be considered, e.g. joint limits. We have found that most of them can be treated by the time to contact idea described above for obstacle avoidance. For example, it is possible to calculate the time at which the joint limit is reached for each movement direction.

Simulation of manipulator control

In the simulation presented here, the task is to move the endeffector from an initial position at the robot's right side to a target position depicted by a "T" (see fig.8). The straight way is blocked by an obstacle located in front of the robot. The endeffector orientation is kept constant. To fulfill the task the elbow moves in an upper position.

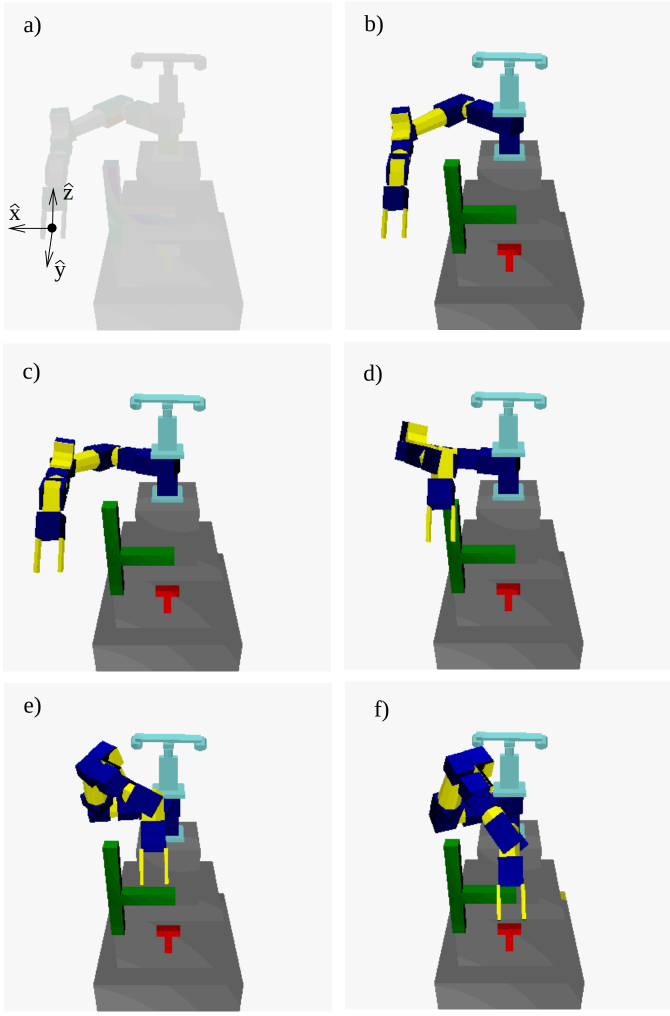

Fig. 8. Movement sequence

Due to limited space we only discuss the endeffector's positioning:
Figure 9 shows activation (left columns) and stimulus (right column) of the 'neural sphere' described above.

Let us define an absolute endeffector centered coordinate system in which x is oriented on the right, y ahead, and z up (fig.8(a)). These vectors represent points on the movement direction sphere and are depicted in the stereographical projection of figure 9 by points on a circle (fig.9(a)).

At the beginning (fig.8(b)) the target is represented by an activity peak located ahead and left of the robot's endeffector, whereas the obstacle, located on the endeffector's left (black circles in stimulus plot (fig.9(b))) occludes part of the target stimulus. The neural field acts as a decision making mechanism and "chooses" among the possible directions (right, ahead/back, up/down) the best movement to perform : ahead, up (bright circles in activity plot fig.9(b)). In the step snapped by figure 8(c), obstacle is represented down,left and the best movement is choosen to be left, ahead, and up. In the next step (fig.8(d)) obstacle is represented left, down and the best movement is

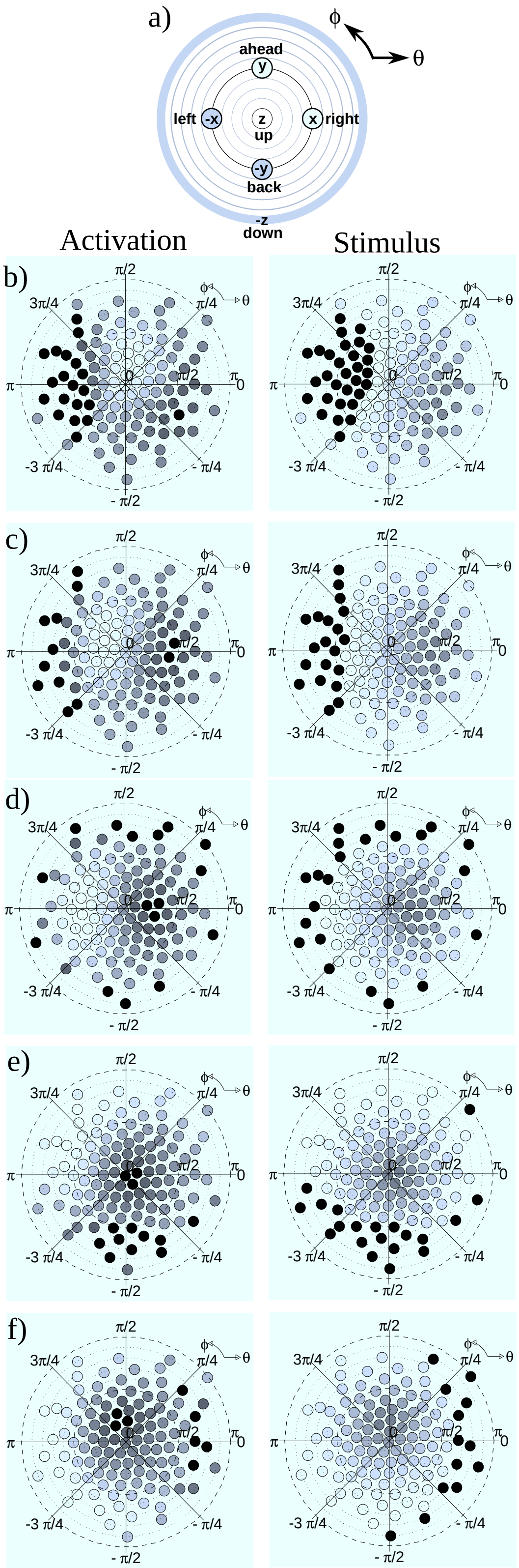

Fig. 9. The diagrams depict a stereographical projection of the sphere's surface (see text), where the center represents the north pole, and the outer bound the south pole.

left, up. Then (fig.8(e)) obstacle is represented back, down and the best movement is left, ahead and down. Finally, (fig.8(f)) obstacle is represented right, down and the movement "choosen" by the neural field (left, back, down) allow to reach the target. One should note that the elbow dynamic

which is not depicted here was also needed for the simulation.

V. Conclusions

We presented a new approach to robot motion control. In applications to local navigation and manipulation we could show that it is well suited to solve the problem of reaching a goal under constraints:

- movement directions are represented as codebook vectors, which can be learned by a neural gas algorithm, for example.
- the intrinsic dynamical properties of neural fields lead to stable selections of appropriate movement directions, even in the case of ambigious situations.
- goal and constraints can be represented in an easy fashion.

Since our application to local navigation has already been implemented on our robot 'ARNOLD', our future work focusses on the implementation of the manipulator control decribed here.

References

[1] Shun-ichi Amari. Dynamics of pattern formation in lateral-inhibition type neural fileds. *Biological Cybernetics*, (27):77–87, 1977.

[2] E. Bicho and G. Schöner. Target position estimation, target acquisistion and obstacle avoidance. In *Proceedings of the IEEE International Symposium on Industrial Electronics, ISIE'97*, 1997.

[3] Percy Dahm and Carsten Bruckhoff. Autonomous decision making in local navigation. In *From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior (SAB 98), (to appear)*. MIT Press, 1998.

[4] Percy Dahm and Frank Joublin. Closed form solution for the inverse kinematics of a redundant robot arm. Technical Report 8, Institut für Neuroinformatik, Ruhr-Universität, Universitätsstraße 150, D-44780 Bochum, Germany, April 1997.

[5] Christoph Engels and Gregor Schöner. Dynamic fields endow behavior-based robots with representations. *Robotics and Autonomous Systems*, 14:55–77, 1995.

[6] A. P. Georgopoulos. Cognitive neurophysiology of the motor cortex. *Science*, 260:1–132, 93.

[7] M. A. Giese, Schoener G., and H. S. Hock. A dynamic neural field model for the perceptual organization of apparent motion. *nvestigative Ophtalmology and Visual Sciences*, 37:745, 1996.

[8] M Jeannerod. *The Neural and Behavioural Organization of Goal-Directed Movements*. Oxford Psychology Series. Oxford Science Publications, 1988.

[9] O. Kathib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.

[10] K. Kishimoto and S. Amari. Existence and stability of local excitations in homogeneous neural fields. *Journal of Mathematical Biology, Springer*, 7:303–318, 1979.

[11] Schulten K. Martinetz T. A 'neural-gas' network learns topologies. In *ICANN 91*, pages 397–402. Elsevier, 1991.

[12] G. Schöner, M. Dose, and C. Engels. Dynamics of behavior: theory and applications for autonomous robot architectures. *Robotics and Autonomous Systems*, (16):213–245, 1995.