To appear in :Applications of Modern Heuristic Methods: Neural Networks, J.G.Taylor (ed), Unicom & Alfred Waller Ltd. Publ.

Route Finding by Neural Nets

Guido Bugmann, John G. Taylor* and Michael J. Denham

School of Computing, University of Plymouth, Plymouth PL4 8AA, United Kingdom
Phone (0752) 23 25 66; FAX (0752) 23 25 40; email: gbugmann@soc.plym.ac.uk
* Department of Mathematics, King's College London, London WC2R 2LS, United Kingdom

Abstract
This paper describes a neural implementation of the resistive grid technique for route finding. The resistive grid, or Laplacian planning technique, is not plagued by local minima problems and guaranties and existing route to be found. The neural network comprises 2 layers. In the upper layer, lateral connections between neurons communicate information on the potentials of neighbouring nodes in the grid. The lower layer represents a spatial memory in which information on the positions of obstacles an the target is stored. Each neuron in the upper layer receives a single input from a node in the lower layer corresponding to the same spatial location. This input is used to constrain the potentials in selected nodes in the resistive grid. The interplay between resistive grid and spatial memory results in a very flexible architecture easily adaptable to new environments. Its properties are demonstrated with a 2-dimensional path-planning problem for a mobile robot. The Dirichlet and Neumann boundary conditions are compared in terms of routes found and computational costs. Limitations and possible developments of the resistive grid technique are discussed.

1) Introduction

Route finding is the problem that a mobile autonomous robot has to solve when required to move from a starting point to a destination point, in an environment cluttered with obstacles. This problem can be generalised to N-dimensional state spaces. It is then stated as: Having a system in a given initial state, what sequence of actions will allow the system to reach a preset final state, knowing that a number of states are forbidden. In the case of the mobile robot, the state-space is the 2-dimensional position space and forbidden states are the positions occupied by obstacles.

A variety of problems can be defined as route finding, or path-planning, in an N-dimensional space. For instance balancing a pole [Barto et al., 1983] where, for whatever state the system is in, one must find the action leading to a more balanced condition. In chess or backgammon playing [Tesauro, 1991, and references therein], a sequence of actions must be generated which leads to a winning position. In planning some future profession, a sequence of intermediate steps has also to be planned.

Various methods have been developed to solve these problem, but all share a common principle: the use of an evaluation map. An evaluation map is a set of values attached to states, the value being a decreasing function of the distance between that state and the goal state. Once the evaluation map has been constructed, route finding becomes an easy task. It consists of finding which neighbouring state has a higher evaluation than the current state, then performing the action causing the transition to that better state, then searching again for the best next state. This ensures the goal to be eventually reached.

The use of evaluation maps transforms the problem of route finding into a problem of finding the values of the states. We will describe in section 2 how this can be done very simply using the resistive grid method. In section 3 we describe a neural implementation of the resistive grid which allows a great flexibility in dealing with changing environments. The examples of the control of a mobile robot will be described in section 4. In section 5 we discuss variations and extensions of the method, its advantages and weaknesses, and a possible biological implementation. Section 6 is the general conclusion.

2) The Resistive-Grid Method

In the resistive grid method, also called Laplacian path planning [Connolly et al., 1990], the state space is divided into a set of small *N*-dimensional cubes. Each cube corresponds to a node in a resistive grid. Each node is connected to its 2*N* nearest neighbours by 2*N* resistors *R* (figure 1). Nodes on the edges and corners have a smaller number of neighbours. The same value *R* is used in the whole network. When using such a grid for route finding, the node corresponding to the target state is usually set to a positive potential, for instance 1. This node acts as a current source.

The epithet "Laplacian" and the later epithet "Dirichlet" and "Neumann" arise from the fact that when the potential distribution in the grid is calculated numerically, the update equations, in the limit of many nodes in the grid, become those of the heat equation with dimension equal to that of the grid.

To see this, we note that the update equation for the potential y_i, for the i^{th} node, where that node is at the position $\underline{r}$ and its 2*N* nearest neighbours at $\underline{r}+\Delta\underline{r}_l$ (l =1,..,2*N*) is

$$y_i(\underline{r}, t+\Delta t) = gI_i + \frac{1}{2N}\sum_{l=1}^{2N} y_i(\underline{r}+\Delta\underline{r}_l, t)$$

where I_i is any external current and g the input resistance to the node. For small Δt, this equation becomes

$$\dot{y}_i(\underline{r}, t) = \frac{g}{\Delta t} I_i + \frac{1}{\Delta t}\frac{1}{2N}\left[\sum_{l=1}^{2N} y_i(\underline{r}+\Delta\underline{r}_l, t) - y_i(\underline{r}, t)\right]$$

The term in the square brackets, in the limit of the distance to the nearest neighbour being infinitesimal, is the *N*-Dimensional Laplacian ∇^2 applied to y_i , and the above equation therefore reduces to the heat equation

$$\dot{y} = aI + k^2\nabla^2 y$$

where the coefficient of diffusion k is $(\Delta r^2/2N\Delta t)^{1/2}$. The static limit of this equation reduces therefore to the Poisson equation

$$\nabla^2 y = -bI$$

The heat and Poisson equations are only well defined provided suitable boundary conditions are supplied. In the Dirichlet case, these correspond to specifying y on the spatial boundary B of the domain. The Neumann conditions are determined by giving the value of the normal derivative of y on the boundary B. Either of these sets of boundary conditions are known to give an unique solution to the Poisson equation for suitable conditions on the source current I.

Accordingly, when using the resistive grid for path planning, there are 2 variations on how to define the boundary conditions applied to obstacles (forbidden states) and the potential set at the node in the grid corresponding to the current state:

a) In this first variation, the nodes corresponding to obstacles have their potential set to a potential zero. This corresponds to the Dirichlet boundary condition which was first proposed by [Connolly et al., 1990]. With this condition, an electric current flows from the target node towards all obstacle nodes, acting as current sinks. Whatever the current state of the system, is sufficient to move towards the neighbouring node with the highest potential to eventually reach the target state. In this variation, the potential distribution is fixed for a given configuration of target and obstacles, and does not depend on the current state of the system.

b) In this second variation, the obstacles are assumed to constitute a non-conducting material. This corresponds to the Neumann boundary condition proposed by [Tarassenko and Blake, 1991]. In practice, this is achieved by interrupting the connections to the obstacle nodes. In this variation, the node corresponding to the current state is used as the current sink and its

potential is set to zero. Thereby, the potential distribution in the grid changes continuously, as the system evolves from state to state.

In both variations, for the current flowing out of the target node to be able to reach the current-state node, there must exist an uninterrupted sequence of permitted states joining the target and the current state. Therefore, the resistive grid method garanties that the target will eventually be reached. With the Dirichlet boundary conditions, the potential decreases exponentially as the distance of the target increases. Therefore, one must be able to read very small gradients to use this method in practice. This problem does not arise with the Neumann boundary conditions [Tarassenko and Blake, 1991]: The potential is very flat everywhere except in the immediate neighbourhood of the current state (sink), where large gradients can be used to determine the direction of the next move. Due to different distributions of current lines, the two variations lead to slightly different route, as will be illustrated in section 4.2 for a 2-dimensional case.

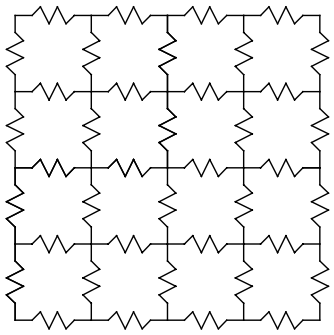

Figure 1:
2-Dimensional resistive grid with 5 x 5 nodes.

3) Neural implementations of a resistive grid

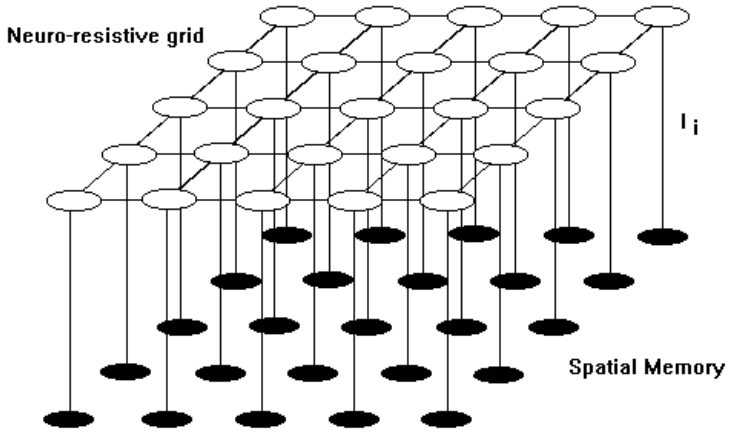

Figure 2:
Neural network implementing a 5 x 5 resistive grid and the spatial memory layer used to constrain the activities of nodes in the resistive grid corresponding to the target and the obstacles positions.

Each node in the grid is represented by a neuron which receives inputs from its *m* neighbours and from one neuron in a "spatial-memory" layer (figure 2). Each neuron *i* calculates its output, or its potential y_i as follows

$$y_i = Tf(\sum_{j=1}^{m} W_{ij} \cdot y_j + I_i)$$

where W_{ij} is the weight given to the input from neuron j to neuron i; y_j is the output of neuron j; I_i is an external input used to constrain the value of y_j, and Tf is the transfer function of the neuron i. We use a linear saturating transfer function illustrated below in figure 3.

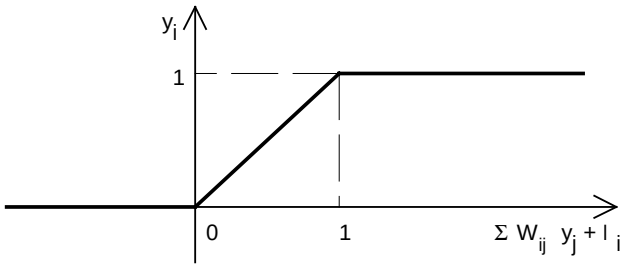

Figure 3:
Linear-Saturating transfer function of the neurons representing nodes of the resistive grid.

Various other forms of transfer functions can be used, provided they satisfy two conditions:
i) Their output must be 0 if the input is 0;
ii) Their gain, or product of their steepest slope time m W_{ij} , must be smaller or equal to 1.
The condition i) is not satisfied by the classical sigmoid transfer function used in most artificial neural networks: $y_i = 1/[1+\exp(-\sum x_j w_{ij})]$. With this sigmoid transfer function, neurons become current sources and the potential in the network can develop local maxima which prevent the goal to be reached. This point has been overlooked in [Glasius et al., 1994]. If the gain is larger than 1, the network is unstable and may end in a state where all neurons reach the saturation potential 1. Thereby, there would be no potential gradient to exploit for route finding.

By using $W_{ij} = 1/m$ (= 0.25 in a 2-Dimensional case), the neuron sets its potential to the average potential of its m neighbours. This is exactly how nodes set their potential in a real resistive grid. As nodes at the edge of a 2-Dimensional grid have only m=3 neighbours and those at the corners only m=2 neighbours, their inputs weights must be set to W_{ij} =0.333 and W_{ij} = 0.5 respectively. The saturation of the transfer function for inputs larger than 1 is only necessary for the target neuron, because we set its potential to 1 by adding an external input I_i= 1, so that its output is y_j =1, whatever the potentials of the neighbours in the grid. In the Dirichlet case, the saturation for negative inputs allows us to set to 0 the potential of nodes corresponding to obstacles, by using a negative input I_i = -1. In the Neumann case, only the potential of the node corresponding to the current state is set to 0 with a negative input I_i = -1 and the weights W_{ij} must be set to zero when there is no path between the states i and j. This would be the case if state j represents an obstacle. We may note that, when the number of inputs is reduced by disabling connections, the values of the input weights W_{ij} must be adapted to the new number of inputs. If this is not done, missing inputs act as sinks, because they reduce the calculated average potential. In both conditions, for all nodes which are neither targets, current positions nor obstacles, I_i = 0 and the nodes determine their potential freely, according to the potentials of their neighbours.

All neurons in the network must be updated several times before an equilibrium distribution of the potentials is achieved. Theoretically, an infinite number of updating cycles is needed in order to reach the same distribution as in a real resistive grid. In practice, correct directions of gradients are achieved after a few tenth iterations. The absolute minimum number of iterations is given by the maximum number of nodes in the path from the target to any node in the grid. In a complex maze, for instance, a quite large number of iterations may be needed before the information on the potential at the target reaches the current position of

a robot. Using the Neumann boundary condition, the potential distribution must be recalculated after each change of state. In the Dirichlet case, the distribution can be calculated once for all, as long as obstacles or the target are not moved.

The neural implementation of the Neumann condition is more complicated than the Dirichlet condition. In the Dirichlet condition, it is sufficient to set activities in nodes in the spatial memory layer. In the Neumann condition, one must also modify weights in the resistive grid, setting to zero those corresponding to forbidden transitions between two states and recalculating the values of active weights to ensure a correct emulation of the behaviour of a resistive grid.

In the case of path planning in a 2-Dimensional maze with n x n positions, the Neumann approach seems more economical in terms of number of nodes in the resistive grid, because the thickness of walls is negligible. With the Dirichlet approach, extra nodes would have to be used for the walls and one would end up with $(2n-1)\text{x}(2n-1)$ nodes, approximately 4 time more than in the Neumann case. However, the complexity of the extra circuitry needed to control the weights in the Neumann approach may outweigh the advantages of having less node in the resistive grid. In simulations one does not need to construct that extra circuitry explicitly, an algorithmic part of the program could emulate its function. However, the resistive grid needs to be updated several times after each displacement of the robot, in order for the resistive grid to reach its new equilibrium distribution of potentials. Therefore, the Neumann approach involves more computation time than the Dirichlet approach, at least in the neural implementation. Hardware implementations of the Neumann approach are currently developed, which may overcome these problems [Marshall and Tarassenko, 1994]. In hardware, weights need no to be recalculated when links are cut and new equilibrium distribution of potentials are rapidly reached.

We have investigated the possibility that a single calculation of the potential distribution, with the starting state as sink, may suffice to determine an adequate path to the target. However, by simulating this condition in a 2-Dimensional case, we have found that the robot had the tendency to follow current lines along obstacles. For instance, in the configuration of figure 5, the robot followed the inside of the h-shape and then the outside up to very near to the target. This is obviously not a desirable path. The cause for that behaviour is not clear yet, but it is not observed when the potential is recalculated for each new position. For the time being, it seems necessary to follow the computationally expensive re-updating procedure.

In section 4.1, we will address the practical question of how knowledge about target, current state and forbidden states (obstacles) is fed into the resistive grid. In section 4.2 we will illustrate the differences between the Dirichlet and Neumann conditions with a 2-Dimensional example of a mobile robot in a room with obstacles.

4) Application to a mobile robot

4.1) Knowledge acquisition and exploration

The process of knowledge acquisition is illustrated in figure 4. A robot has to move in the simulated room divided in 20 x 20 squares. The robot has the size of one of the squares. Obstacles forming a maze must be avoided by the robot in order to reach a target in the upper part of the room. The robot is provided with a scanner giving the distance of obstacles over a limited range. The obstacles are not transparent and only those in direct line of sight can be detected. The robot is also supposed to have the information of the position of the target. In what follows, we will describe the implementation in the Dirichlet case.

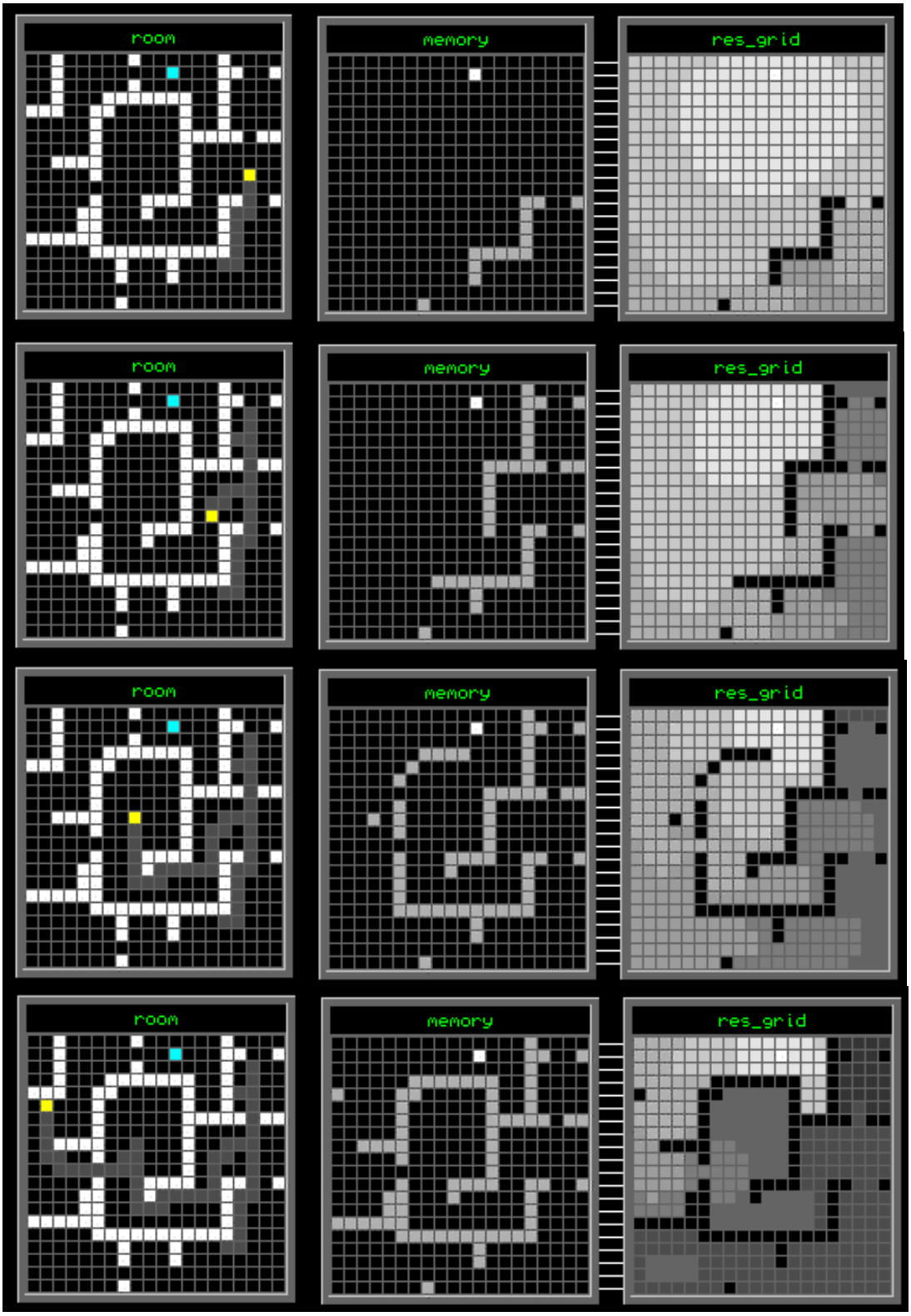

Figure 4:
Examples of paths followed by a mobile robot with an initial empty memory.
The frame on the left represents the room. The target is the grey square bellow the "m" of "room". White squares are obstacles. The grey track is the path followed by the robot, which is located at the end of the track. The frame in the middle is the spatial memory.
The frame on the right is the neuro-resistive grid. The brighter the squares, the higher the potential of the corresponding node. Following the figures from top to bottom, one

can observe the increasing knowledge of the obstacle configuration placed in spatial memory.

The spatial memory is a grid of 20 x 20 neurons the outputs of which can be set to +1, 0 or -1. When an obstacle is detected, the output of the neuron in the spatial memory corresponding to the same position is set to I_i = -1. This implies implicitly that the robot knows where he is in the room. That point will be discussed further in section 5. The neuron corresponding to the target has its output set to I_i = +1. The nodes corresponding to free space have an output I_i = 0.

The resistive grid comprises 20 x 20 neurons receiving inputs from their *m* nearest neighbours and from one neuron in the spatial memory corresponding to the same position. The content of the memory constrains the outputs of the neurons in the resistive grid, as described in section 3. The network is simulated on a PC 486DX using the simulation package CORTEX-PRO.

The robot starts usually in the lower right part of the room with no knowledge of the position of the obstacles. A first scan is done and the found obstacles placed in memory. The resistive grid is then updated 40 times and the resulting potential used for determining the 10 next moves during which no scan of the environment is performed, to reduce computation time. If, the robot hits an obstacles during these 10 "blind" steps, a new scan over a very short range is initiated, which allows to store in memory the topography in the immediate neighbourhood of the hit obstacle. In figure 4 one can see how the knowledge of the robot progressively increases as he moves towards the goal. Before the existence of obstacles is known, the memory is empty at the corresponding locations and the potential in the resistive grid builds up as if there was free space in place of the obstacles. Therefore, the initial movements of the robot are directed towards the goal. For instance, in figure 4. the robot has to approach the top right hand-side corner of the room to find out that it is a dead end. Once the knowledge of the robot is complete, and if he is restarted at the same location, he chooses a direct path towards the target.

There is no exploration behaviour explicitly built-in the robot, but the initial attempts to reach the target with incomplete knowledge lead to the acquisition of spatial knowledge, in a way similar as during exploration. This however not equivalent to spontaneous exploration, as observed with animals, e.g. rats, placed in new environments [O'Keefe and Nadel, 1978]. While the drive of our robot is to reach the target, animals seem to have a need for spatial knowledge.

4.2 Comparison of Dirichlet and Neumann routes

The path followed by the robot is not necessarily the shortest, because it follows the current lines, but it is smooth and avoids obstacles. In the Dirichlet condition, obstacles act as current sinks and current lines are initially perpendicular to obstacles. When a robot starts near to an obstacle, it will therefore take a trajectory which first increases the distance to the obstacle, and then approaches the target. In the Neumann condition, obstacles are invisible to the current flowing in the resistive grid, and the current lines are parallel to the surface of obstacles. Therefore, path found with the Neumann approach are have less clearance from the obstacles. In figure 5, we show the difference in direction of the potential gradient in the two cases, and the resulting trajectories.

We have chosen the particular obstacle configuration in figure 5 because, using former "potential field methods" [see references in Connolly et al., 1990], the robot would get stuck in local minima. With these methods, the target is the source of an attractive field and the obstacles generate a repulsive field. There are two points in figure 5 where the robot would be equally attracted by the target and repulsed by the obstacles, and would stop moving.

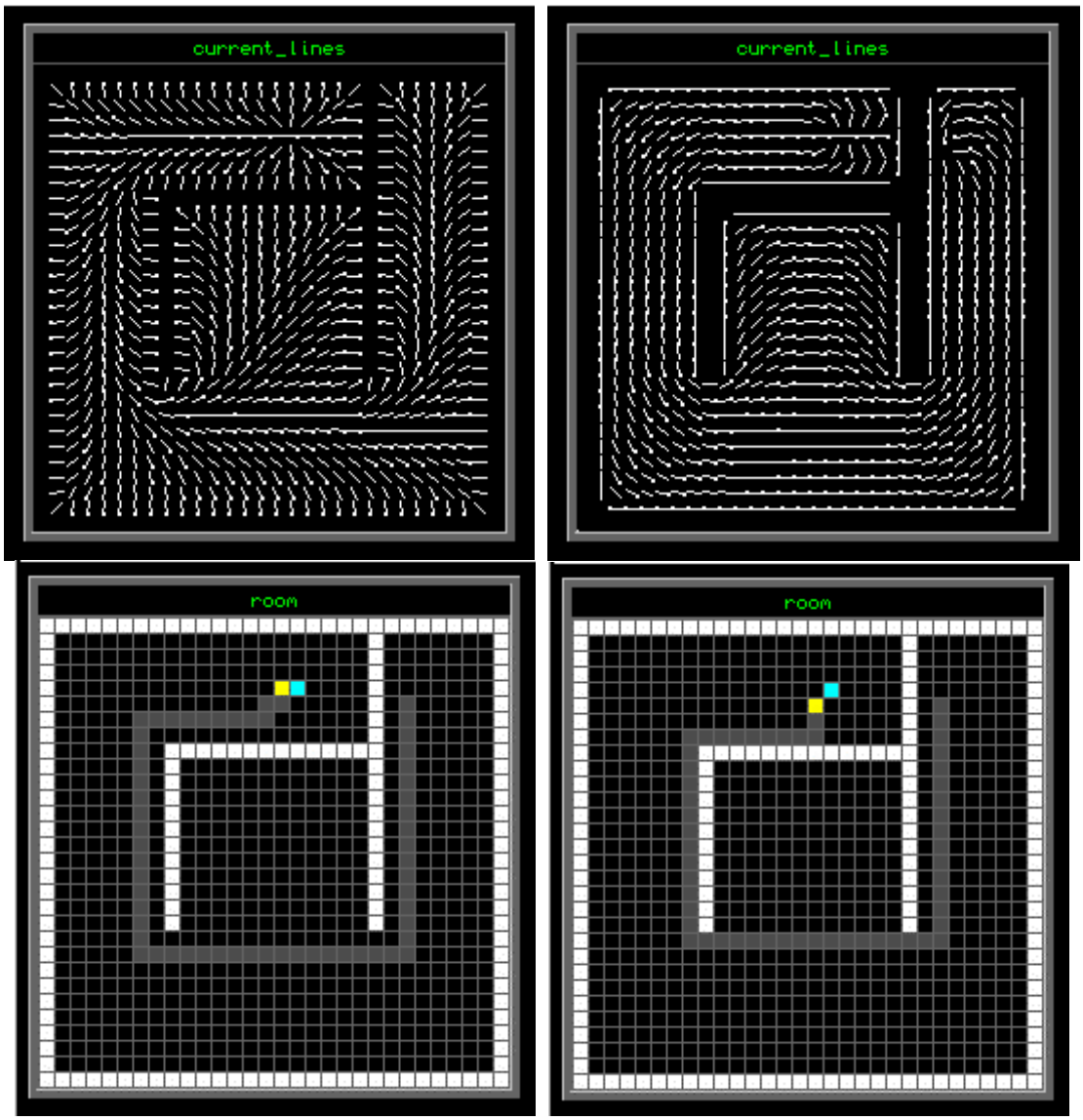

Figure 5:
Top row: current lines in the resistive grid for the Dirichlet condition (left) and the Neumann condition (right). In the Neumann condition, the currents are calculated for the robot placed in the initial position. Bottom row: Trajectories of a robot according to the Dirichlet condition (left) and Neumann condition (right). We show the current lines and paths found when once the spatial knowledge is complete.

5) Discussion

Resistive grid:
The resistive grid method can theoretically be expanded to problems with state spaces of high dimensionality. However, in hardware, it is very difficult to realise a resistive grid with dimension larger than 3. The same connectivity problems occur in a hardware realisation of a neural network implementing the resistive grid. In simulations, there are no such connectivity problems. However, one faces rapidly memory size limitations and huge computation times, especially when high resolution grids are required by the problem.

Another problem is the resolution, while a resistive grid of reasonable size, e.g. 100 x 100 could be used to find routes in rooms of limited dimensions, it is probably not sensible to attempt to navigate in a city using this method. For problems involving large distances, e.g. "how should I do to go from New York to London", one may wish to exploit the good properties of the resistive grid, namely the use of connections between neighbouring states, but re-think the attribution of locations to nodes. Probably, using nodes as landmarks, e.g. "London airport", "train station", "door of my room", would be a good approach. Indeed, the selection of such landmarks is another question, which we do not wish to address in details here.

A first step towards the landmark problem would be to prune the state space in order to keep only relevant states. For instance, in a maze, only positions corresponding to junctions need really to be coded for. Using such a *sparse state space* would however require a more refined coding of the action to be performed to join two states. With equidistant states, the movement to perform is independent on the two positions to join. While with a sparse state space, the action or sequence of actions allowing to join the states are specific to these two states. We may note that this approach where the *relevance for decisions* determines the locations to be represented differs from "adaptive range coding" where the frequency of stays in a given part of the state space determines the density of nodes encoding that part [Rosen et al., 1992].

The inevitable coarseness of the division of a state space into discrete states can also lead to problems when fine control, for instance of a gripping robot hand, is required. A satisfactorily solution of that problem may be found using a grid with *variable node density*, for instance using the technique based on Kohonen maps proposed by [Rosen et al., 1992]. However, the more specialised the partition of the state space, the less flexible the planning system. It therefore possible that more fundamental changes to the resistive grid approach to route finding are required.

Self-localisation:
A hypothesis underlying our simulations is that the robot knows in which position he is. This is needed for placing obstacles found with sensors (egocentric reference frame) into the spatial memory (room-based reference frame). It is also needed to select the node in the resistive grid for which the direction of the current lines has to be determined. In practice it is very difficult for a robot to know its exact location. However, it is possible to infer the position of the robot by comparing the configuration of the currently visible obstacles with the position of obstacles in memory. We have developed a neural network which realises such a task by using a spatial memory operating in polar coordinates [Bugmann, Denham, Taylor, in preparation].

We have not addressed here the question of how the potential gradients can be read and transformed into motor commands using a purely neural network implementation. So far it is done in a purely algorithmic way, but exploratory work indicates that a quite complicated multilayer network may be required. A large part of the complexity arises because of the need to read the gradients for any possible position of the robot in the grid. This has motivated our work towards a resistive grid operating in the egocentric reference frame of the robot [Bugmann, Denham, Taylor, in preparation]. Thereby, the central node in the grid corresponds always to the position of the robot, and the gradients must be read around one node.

Dirichlet versus Neumann condition:
The Dirichlet condition leads to more clearance between robot and obstacles, an advantage when low resolution grids are used. It is easy to implement with a neural network and involves very little computation. Only when new obstacles are found, or old ones removed, must the potential distribution be recalculated. There is no need to recalculate weights in the grid.

The Neumann condition is computationally more demanding. Route are more close to obstacles and may require a grid with higher resolution. In favour of the Neumann condition, it has been argued that the Dirichlet condition leads to impractical small gradients [Tarassenko and Blake, 1991]. So far we have not had any such problems using 32bit real numbers.

Relation to biological systems:
It has been proposed that a part of the brain involved in the control of joint motion, the striatum, may use the resistive grid method [Connolly and Burns, 1993]. These authors proposed that electrotonic link between neurons, i.e. not involving spikes, could implement a resistive grid. As a mechanism for reading out the gradient, a cluster of neurons would first be activated by the current joint configuration, the activity would propagate along the nodes, reaching nodes corresponding to a desired future configuration, and thereby trigger corresponding movements. It is possible, that for the control of joints, which occurs in a finite well defined state space, the resistive grid may prove a good solution. It remains however to be confirmed that the small gradients expected in a large grid can be usefully exploited by biological hardware. On the other hand, as the joint state space has a high dimensionality, and the striatum is essentially a 3-Dimensional network, we may learn useful lessons from the brain on how to represent high dimensional problems in networks of lower, tractable, dimensionality.

However, if the effective lateral connections in the striatum are synaptic rather than electrotonic, a difficulty arises in the interpretation of the action of the striatum as a resistive grid. This is due to the predominant inhibitory action of striatal neurons (containing the inhibitory neurotransmitter GABA). If there were nearest neighbour updating according to the resistive grid model, the resulting heat equation (deductible along the lines of the arguments in section 2) would have a negative value for the constant k^2. "Bunching" would thereby develop, with resulting global pattern formation. The resultant activity is not of clear relevance to route following, although may be of interest in discussions of global control [Taylor and Alavi, 1994].

In the motor cortex, a population of neurons has been found which shows a close relation between its activity and the direction of a movement being performed or planned [Georgopoulos et al, 1984]. In short, each neuron fires maximally for a given direction of the movement. By recording the firing rate of each of these neurons, one can computes a vector which defines a direction of movement which matches the direction of the actual movement performed by the animal. It is interesting to note that we can perform the same operation when comparing the activity of the four neighbours of the current-position node. This is indeed no evidence that the brain uses a resistive grid, but there is an intriguing parallel.

6) Conclusion

The resistive grid method for route finding is very attractive, because it shows no problems of local minima, and ensures an existing path to be found. It is realisable with analogue hardware and can be very fast. The proposed neural implementation of the Dirichlet condition is simple, computationally inexpensive and very adaptive. New obstacles or room configurations can be processed by the same network, simply by changing the content of the spatial memory.

The resistive grid method needs to evolve on several fronts. The extension to problems of dimensionality larger than 3 will require some serious thinking about the question of representations, possibly on ways to link together several low dimensional grids. The question of how to represent large spaces with grids of limited size, may involve some re-thinking of which positions in the real world need to be attributed to nodes in the grid. The

concept of sparse spatial coding may have to be developed. In view of the intrinsic good qualities of the method, the effort is worth doing.

References:

Barto A.G., Sutton R.S. and Anderson C.W. (1983) "Neuronlike adaptive elements that can solve difficult learning control problems", IEEE Trans. on System, Man and Cybernetics, SMC-13, 834-846.

Connolly C.I., Burns J.B. and Weiss R. (1990) "Path planning using Laplace's equation", Proc. IEEE Int. Conf. on Robotics and Automation, 2102-2106.

Connolly C.I. and Burns J.B. (1993) "A model for the functioning of the striatum", Biol. Cybernetics, 68, 535-544.

Georgopoulos A.P., Kalaska J.F., Crutcher M.D., Caminiti R. and Massey J.T. (1984) "The representation of movement direction in the motor cortex: Single cell and population studies", in Edelman G.M., Gall W.E. and Cowan W.M. (eds) "Dynamic aspects of neocortical function", John Wiley & Sons, N.Y.

Glasius R., Komoda A. and Gielen S. (1994) "Neural network dynamics for path planning and obstacle avoidance", Neural Networks, in press.

Marshall G.F. and Tarassenko L. (1994) "Robot path planning using VLSI resistive grids", to appear in IEE Proceedings-F.

O'Keefe J. and Nadel L. (1978) "The hippocampus as a cognitive map", Clarendon Press, Oxford.

Rosen B.E., Goodwin J.M. and Vidal J.J. (1992) "Process control with adaptive range coding", Biological Cybernetics, 66, 419-428.

Tarassenko L. and Blake A. (1991) "Analogue computation of collision-free paths", Proc. IEEE Int. Conf. on Robotics and Automation, Sacramento, 540-545.

Taylor J.G. and Alavi F. (1994) "A global competitive neural network", Biol. Cybernetics, to appear.

Tesauro G.J. (1991) "Practical issues in temporal difference learning", IBM Research Report RC 17223 (#76307), IBM Research Division, Yorktown Heights, NY 10598, USA.