University of Plymouth

School of Computing

Artificial Vision for Micro-Mouse

Vincent Onillon, Guido Bugmann, Alan Simpson and Peter Nurse

Research Report ***NRG-95-05*** July 1995

Neurodynamics Research Group

Artificial Vision for Micro-Mouse

Vincent Onillon, Guido Bugmann[1], Alan Simpson[2] and Peter Nurse[3]

Ecole Supérieure d'Electronique de l'Ouest, Angers, France
[1]School of Computing
[2]School of Electronic Communication and Electrical Engineering
[3]School of Manufacturing, Material and Mechanical Engineering
University of Plymouth, Plymouth PL4 8AA, United Kingdom

Abstract

Vision is important for future robots. This report describes an application of vision to the maze-navigation problem for a micro-mouse. Hardware limitations led us to explore the possibility of using only three points per video line and only 120 lines per image (3 x 120 bytes). Algorithms are described which are able to extract following information from this limited image: i) Orientation of the mouse, ii) Lateral position of the mouse, iii) Distance from the centre of the current cell, iv) Positions of walls up to two cells ahead. The algorithms are tested using a real micro-mouse linked to a PC on which the software is run. Two tasks have been successfully completed so far: i) Driving along a straight row of cells and ii) turning at an intersection. These results show that a micro-mouse can be controlled using only 360 byte of information extracted from a video image.

1. Introduction

Future autonomous mobile robots will need artificial vision for navigation and object manipulation. As the required hardware (camera and microprocessors) is becoming less expensive, it is now possible to investigate the integration of vision and robotics in simple robots like the micro-mouse.

A micro-mouse is a fully autonomous small mobile robot measuring approximately 20 x 20 x20 cm which must navigate in a maze comprising 16 x 16 cells and plan a path to the centre [1]. For that purpose, the micro-mouse must detect and memorise the position of the walls of the maze. As the walls are white and the floor black, one may expect the vision problem to be simple enough to be tackled with limited on-board computing resources.

Due to the computation power and memory size limitations we have explored the possibility of exploiting only 3 pieces of information per line of the video image. These are 1) the time of the last transition from white to black before the centre of the image, 2) the intensity at the centre (black or white) and 3) the time of first transition to white after the middle of the image. In section 2 we will describe how this information is analysed in order to locate the mouse and the walls.

Although this approach to micro-mouse vision is implementable in hardware, we have initially simulated it using a frame grabber and a program running on a PC linked with an umbilical cord to the micro-mouse (figure 1). The main purpose of these simulations is to test if all possible situations that the mouse might encounter can be solved using limited visual information as described above. In section 3 we report on tests consisting of following a straight corridor and turning at an intersection.

In section 4 we discuss the approach and suggest some line of future work towards a competitive visual micro-mouse.

[1]To whom correspondence should be addressed.

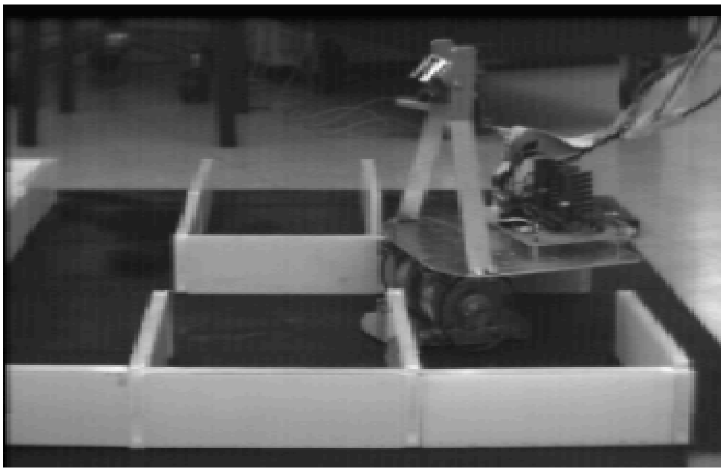

Figure 1. Micro mouse with video camera VVL 1011c-007 Mini Peach with wide angle lens F4.3 (Angle of view 78 deg. horizontal). A cell measures 18 x 18 cm. The walls have a height of 5 cm and a thickness of 1.2 cm. The open space between walls measures 16.8 cm.

2. Vision algorithms

2.1 Reduced data set

Although our current system comprises a frame grabber and a 486-PC, we have processed and pre-processed the data in a way that is compatible with a fully on-board hardware implementation. The duration of a video image frame is 20 ms. It is hoped that, by minimising the number of image points processed per frame, the data collected during one frame may be processed during the next one or two frames.

The composite video signal provided by the camera has 240 lines and a resolution of 310 pixels/line. The important points in the image are the high contrast transitions between walls (white) and floor (black). We have tested that high pass filtering and thresholding allows for robust detection of these transitions. This process is generally unaffected by variations in lighting conditions and can be achieved with a simple hardware circuit.

Assuming a mouse more or less aligned with a row or a column of the maze, the two important transitions are the first ones to the left and to the right of the line of sight (centre of the image). We have assumed a hardware system where a timer with 1 μs time-steps is started at the beginning of each line. A detected high contrast point along a line would activate the temporary storage of the corresponding time. When the middle of the line is detected, the last noted transition would then be stored in the memory of a microprocessor. Similarly, the first transition after the middle of the line would be stored. Storing the current value of the timer is in effect storing the x-position of the transition point.

As the transitions detected along lines only inform on objects generating horizontal contrast, the position of a wall in front of the mouse may not be detected. To have access to vertical contrast information, we have decided also to store intensity information from the centre of each line.

As a video line lasts 64 μs, with 52 μs of image signal, and using a timer with 1 μs resolution, 6 bits are sufficient to store the position of a horizontal transition. Using two bytes per line for the transitions and one byte for the level, an image of 240 lines, requires the storage of only 720 bytes. All the results presented here were obtained using only alternate lines (360 bytes).

The figure 2 shows the reduced transition data set extracted from a typical image. The question which must be answered now is: Can one extract the required information from the reduced data set ?

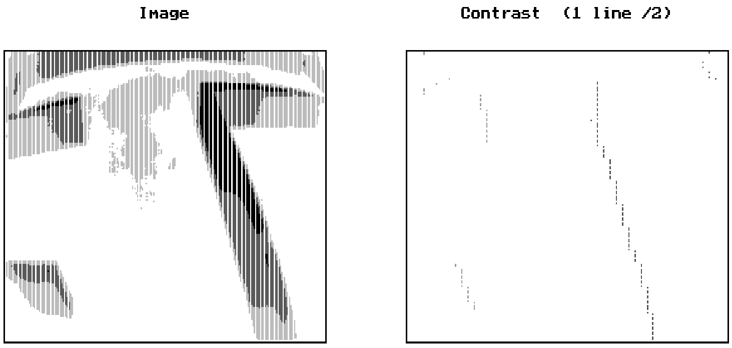

Figure 2. The image to the right shows the points extracted from the image to the left for further processing. The effect of the sampling of a line in 1 µs time steps appears here: a wall is seen as a series of vertical bars.

2.2 Computing the direction

The direction is computed in four steps: A) Elimination of irrelevant data points B) Fitting two straight lines to the data on each side of the centre, C) Computing the camera direction from each line and D) Weighted combination of information from both sides.

A) *Elimination of irrelevant points*: This is done by scanning the data from the lowest line upwards and by eliminating any point which has not a horizontal deviation of exactly 1 time step from the point in the line below. A deviation to the left is required for transition points in the right half of the screen and vice-versa. For instance, in figure 2 (right) this process eliminates the points related to the intersections in the back of the image and a number of redundant points forming short columns due to quantisation effects. The remaining points are shown in figure 3 (left).

B) *Fitting straight lines*: The least mean squares method is used to determine the lines of the form $y_R = a_R + b_R x$ and $y_L = a_L + b_L x$ fitting the remaining points (as above) on each side of the centre of the image. x has values between 0 and 52 (µs). y has values between 0 and 240 (line) where line 240 is at the top of the picture. The resulting lines are shown in figure 5 (top-left).

C) *Computing the direction*: Either of the two fitted lines $y_R(x)$ and $y_L(x)$ can be used to compute the direction d of the camera. We know that for a camera with a given attitude to the horizontal, the full width of each line of the image corresponds to a given width in centimetres at the ground level (the maze). In practice we use only the lines 60 and 160 and measure the number of centimetres/µs for the width of each of these. The distance on the floor between these two lines is ΔL=14.5 cm. Then, for one of the fitted lines $y_i(x)$, we calculate the values of $x_{60,i}$ and $x_{160,i}$ for $y_i = 60$ and $y_i = 160$. Then we convert these values into actual distances from the centre $s_{60,i}$ and $s_{160,i}$. These values are shown in figure 3 (right). The direction is then taken to be the slope:

$$d_i = \Delta L/(s_{60,i} - s_{160,i})$$

This slope enables us to calculate relevant control commands.

D) *Weighted combination*: The quality of the direction information is assumed to be poorer if it is provided by a fit to a smaller number of points. If the number of points used for the fits are m_L and m_R for the left and right line respectively, the average direction is given by

$$d = (m_L + m_R) / (m_L/d_L + m_R/d_R)$$

This expression allows more weight to be given to the direction calculated on the basis of the line fitted with the larger number of points. Moreover, when the image is such that a line can be fitted on one side only, then this expression produces the direction computed from this line only.

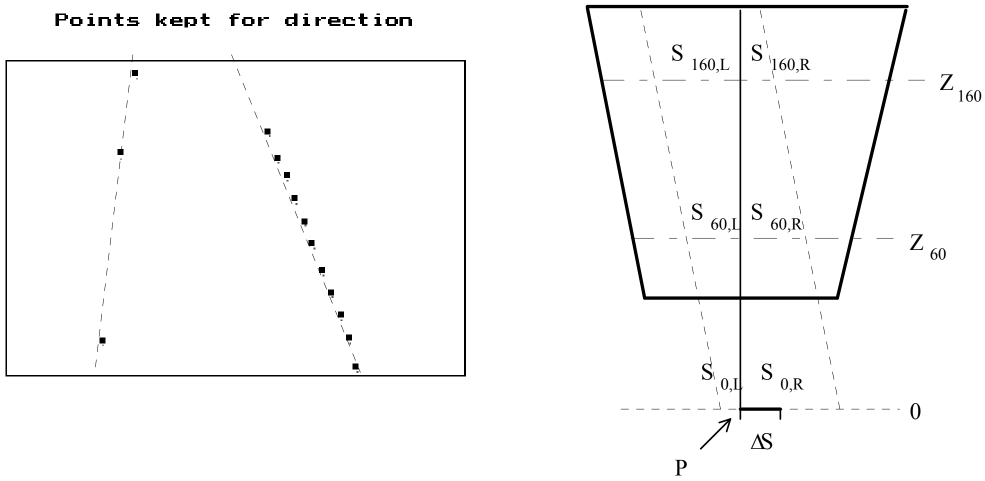

Figure 3. *Left*: Filtered data and fitted lines, as described in section 2.2. *Right*: Top view of a situation similar to the one represented in the left figure. The trapezium in heavy lines approximates the field of view. The position of the mouse is indicated by P. Other symbols are explained in sections 2.2 and 2.3.

2.3 Computing the lateral position

The deviation of the position of the camera from the centre of the cell is calculated in two steps: A) The lateral displacement ΔS at the position of the camera is calculated from each fitted line and B) The weighted average of the lateral displacement calculated.

A) *Lateral displacement*: In the previous section we have calculated the directions d_R and d_L of the right and left edges of the floor in the row. Using the notations from figure 3 (right), and knowing the distances from the camera z_{60} and z_{160} at which the lines 60 and 160 intersect with the ground, one can calculate the distance $s_{0,i}$ from the point on the ground vertically under the camera to the edge of the current cell by:

$$s_{0,i} = (z_{160,}/d_i) + s_{160,i}$$

A displacement of 8.4 cm, half the width of a cell, indicates a centred position.

B) *Weighted average*: The displacement Δs is positive if the camera is in the right half of the cell.

$$\Delta s = [m_L\ (s_{0,L} - 8.4) + m_R\ (8.4 - s_{0,R})] / (m_L + m_R)$$

2.4 Computing the longitudinal position

It is expected that the micro mouse, by keeping a count of the turns of the wheels, can estimate in which cell it is currently located. The purpose of vision is to provide a precise measurement of the distance to the front edge of that cell, in order to recalibrate the internal counting system.

Longitudinal position can be measured by A) first detecting walls in front then, B) openings in the lateral walls then, C) combining information from various sources.

A) *Distance from front walls*: A front wall is located when a sharp increase in stored intensity level is detected along the central line. The line at which this occurs can be translated into a distance Z_j from the centre of the mouse using a stored correspondence table.

B) *Distance from lateral openings*: To determine the position of a lateral opening, the transition data are first reprocessed to eliminate a smaller number of irrelevant points. The difference from the procedure described in section 2.2 is that a point is now kept if it has a horizontal deviation of 1 or 0 time steps with the point in the line below. This allows columns of points to be kept which indicate more precisely the beginnings or ends of lateral walls (see e.g. figure 2, right). A transition wall-opening is detected when a column of points indicating a wall followed by at least 5 video lines with no points. However, an apparent absence of a lateral wall can be caused by the presence of a front wall. Therefore, front walls must be detected first.

C) *Combining information*: The camera is oriented downwards, so that, when the mouse is centred in a cell, the front 2cm of the cell are in the lower part of the image. This enables positional control in dead ends. It also results in a horizontal range limited to 2.5 cells (the camera is 20 cm above ground). In these 2.5 cells there are up to 3 beginnings of cells. So, there are 3 possible front walls to detect and 6 possible lateral wall-no-wall transitions. Only a fraction of these clues is available at any time, depending on the local configuration of the maze. Therefore a weighted average technique must be used to combine automatically distance information from existing clues. Let us name Z_1, Z_2, Z_3,...the distances determined from different clues.

We know that each of these distances is a multiple of the length of a cell (18 cm) added to the distance L to the front of the current cell. So, subtracting from each distance as many times 18 cm as possible, leaves a number of estimates L_1, L_2, L_3,.. of L.

To determine the weights to give to theses estimates in an averaging process, the following argument is used: We have assumed that the mouse may exhibit slight vertical rocking movements which induce uncertainty in the actual distance of a transition located in a given line of the image. This uncertainty increases rapidly as the distance increases (so that it is probably pointless to look more than 2.5 cells ahead). We give therefore a smaller weight to values L_j determined from large values of Z_j. One possibility is to use the inverse of the distances as the weights:

$$L = (L_1/Z_1 + L_2/Z_2 + ...)/(1/Z_1 + 1/Z_2 + ...)$$

Alternatively, one can also calculate the error associated with each line and make a table of appropriate weights.

2.5 Locating walls

As the distance to the edge of the current cell is known, each cell is now well located and the presence or absence of walls can now be checked. Figure 4 (left) illustrates the 13 locations of possible walls surrounding the visible cells. The distances e_i are given by: $e_1=L$, $e_2= e_1 +18$, $e_3= e_2 +18$. To these distances correspond lines ℓ_1, ℓ_2 and ℓ_3 in the video image.

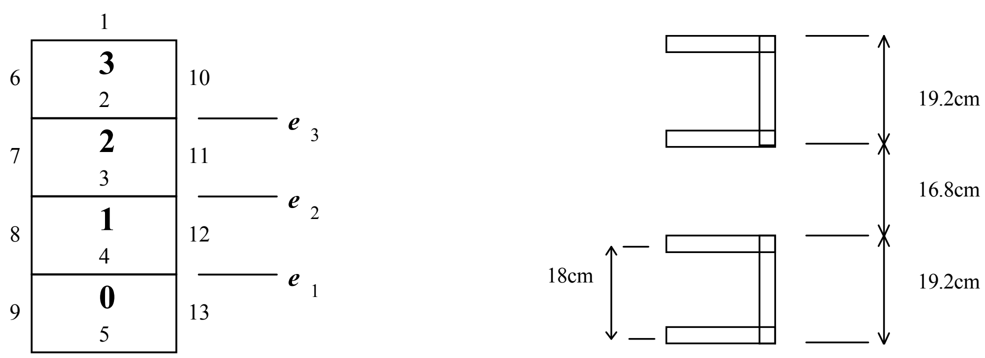

Figure 4 *Left*: Diagram representing the cells that the mouse can analysed when it is in cell 0. *Right*: Example of the effects of wall thickness on the apparent sizes of cells.

The presence of horizontal walls is checked by looking for transition points just above ℓ_1, ℓ_2 and ℓ_3 and the vertical walls on the side are checked by looking just under. When ℓ_3 exists, cells 1 and 2 are fully known (walls 2,3,4,7,8,11,12). When ℓ_3 is beyond the top of the screen, that is when the mouse is situated at the beginning of cell 0, then cell 1 is fully determined and walls 9 and 13 can be checked.

Due to the small number of lines covering a cell at the top of the picture, detection of walls 6 and 10 often give bad results. Walls 1 and 5 are of course never seen.

One may note that, once the position of the walls is determined, a better estimation of the distance L can theoretically be achieved. Figure 4 (right) illustrates the effect of wall thickness on the measured distances Z_1, Z_2, Z_3,.. Therefore, by knowing where walls are, one can subtract the correct number of centimetres to determine the values L_1, L_2, L_3,..used in section 2.4. Another source error in the determination of longitudinal distance is due to orientation effects. When the mouse is not looking straight ahead, the points of intersections on the left and on the right are not on the same video line. This can theoretically also be compensated for using the direction knowledge.

2.6. Results of visual information processing

An example of the results of visual information processing is shown in figure 5.

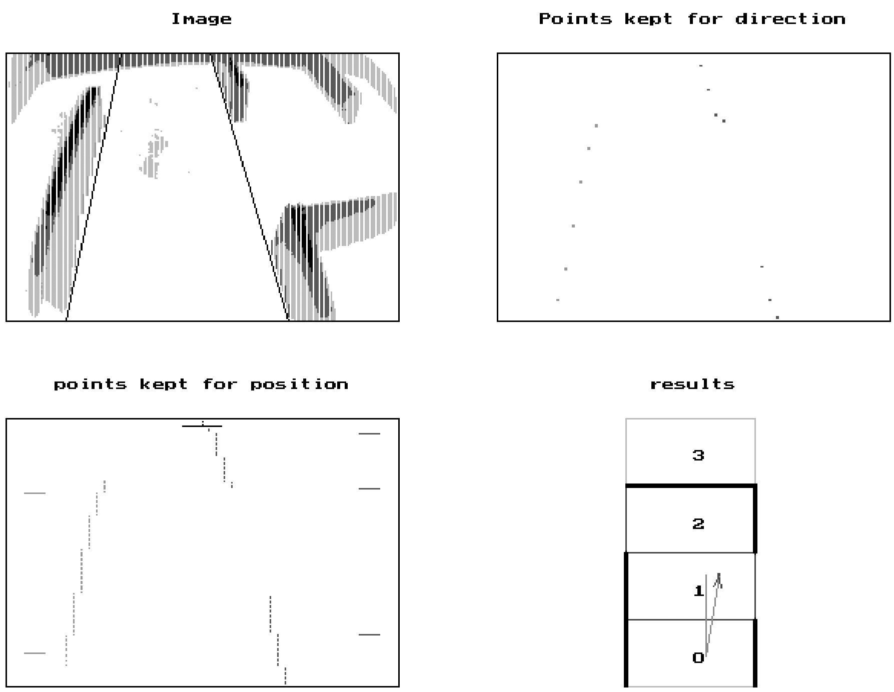

Figure 5. Copy of screen displayed on the PC with results of the processing of the input image shown in the top left frame. ***Top-left***: Input image displayed with 4 grey levels. The two lines are the result of a fitting by the Least Mean Squares method to the points in the top-right frame. ***Top-right***: Points kept for the determination of the direction. ***Bottom-left***: Points kept for the determination of the position and the location of the walls. The sampling effect clearly appears. The horizontal segments on this frame mark the lines noted as indicating beginnings of cells, from left and right intersections and transversal walls. ***Bottom-right***: The final result. Four cells are displayed, the mouse is in the first one (as in figure 4). Its estimated direction is indicated by the arrow. The origin of the arrow is situated at the estimated vertical and horizontal positions of the mouse in its cell. When a wall is detected it appears as a thick line.

3. Tests with a micromouse

3.1 Equipment

The micromouse is of wheel-chair type with two stepper motors driving directly wheels with a diameter of 74mm. The mouse carries a printed circuit board with the drivers for the stepper motors, and a wide angle video camera VVL 1001c-007. The camera provides a composite video signal with 240 lines per frame and a resolution of 310 pixels per line. The composite signal is acquired by a WinVision frame grabber (provided by VVL) mounted in a 486-PC DX-33. The frame grabber has 48K of internal memory and samples the video lines with a resolution of 187points/line. The PC runs a C program comprising a driver for the frame grabber, an image analysis algorithm, a control algorithm and a driver for a I/O card sending step and direction commands to the driver board on the mouse. Image analysis uses typically 100ms of processing time with the current Turbo-C program which is not optimised.

3.2 Following a corridor

This test requires A) A sufficient sideways directional range of the image processing algorithm and B) An adequate control strategy.

A) *Directional vision range*: As the vision algorithm assumes that the right edge of the corridor will appear in the right half of the image and vice-versa for the left edge, there is a limit of directional deviation that the vision system can tackle. To determine this limit, we have placed the mouse in the centre of a cell with a number of different initial orientations and with a number of different maze configurations. We have then checked that the vision algorithm gave the correct direction and position of the mouse.

The results show that the mouse can recognise deviations of up to 10^{o} with an accuracy of $\pm 1^{o}$.

B) *Control strategy*: When the mouse is out of line, or displaced laterally, we first rotate the mouse on the spot to aim at a point 10cm ahead along the centre line of the corridor. Then mouse advances by 10cm and a new picture is taken.

The mouse has been able to keep moving in a corridor measuring 4 cells, with initial direction deviation between $+10^{o}$ and -10^{o} and lateral displacements of ± 1.5 cm.

3.3 Turning at an intersection

The task assigned to the mouse was to turn right at the first intersection. This task requires the detection of an intersection (which has been demonstrated in section 2.6), A) An accurate positioning of the mouse before rotating and B) an adequate rotation strategy.

B) *Positional accuracy*: The test done previously with various positions and orientations of the mouse have shown that, with directional deviations of up to 10^{o}, the lateral position is known within ± 0.5 cm and the longitudinal position within ± 2 cm. The poor longitudinal accuracy is due to the use of simple one-pass algorithm described in section 2.4. The recurrent use of knowledge on the maze configuration, as suggested in section 2.5, is not implemented yet. However, when the longitudinal position is measured with a mouse looking straight ahead, the directional sources of errors noted in section 2.5 are minimised. The longitudinal accuracy is then approximately ± 0.5 cm. As the width of mouse is 4.5 cm smaller than the width of an opening, this accuracy is sufficient to ensure a correct positioning before rotation.

C) *Rotation strategy*: For these initial tests, a single rotation strategy was adopted. The mouse advances to the centre of the cell facing the opening and makes a 90^{o} rotation on the spot. It then takes a picture to determine the next action.

In the test configuration shown in figure 6, the mouse was reliably able to go from the starting point S to the goal G. This involved following five straight lines and four turns.

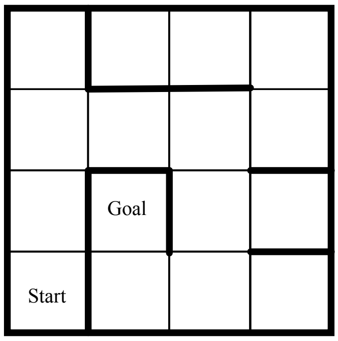

Figure 6. Maze configuration used to test the capabilities of the mouse to detect and turn at intersections.

4 Conclusion

The results of the tests performed so far indicate that by using only 3 pieces of information per line of a video frame, and using only 120 lines per frame, it is possible to extract from the image sufficiently accurate information needed for the control of a micro mouse. Although a number of details need improvement, we are reasonably confident that it may be possible, using limited computing and memory resources, to construct a fully functional autonomous mouse using a video camera as the only sensor.

The approach suggested here should enable the storing of the data extracted from one image in real time (20 ms) and to process it during the one or two succeeding frames (20 to 40 ms). As a fast mouse might advance by more than 4 cm during that time, interesting predictive control problems might be encountered. Before such problems are solved, micro-mice using traditional arrays of infra-red sensors will continue to win competitions.

5 Reference

1. "World Micromouse Competition: Technical Information Pack", This document contains the rules of the competition, an Introduction by Alan Dibley, and a list of references to technical publications. It can be obtained from:

David Penrose (CG)
Computing and Control Division
Institution of Electrical Engineers
Savoy Place
London WC2R 0BL

6 Acknowledgements

We gratefully thank a number of people who have contributed to these results in many ways: Christophe Bianco, Bob Bray, Derek Carter, Phil Culverhouse, John Eastman, Victor Hamon, Gordon Hanley, Marc Hellequin, Sheila Newman, Francois Lobet, Guy Lumeau, Paul Robinson, Brian Pateman, Pat Pearce and Mike Sloman.