Atrium Robot Swarm Controller Specification Vn 2.1

Bob Fisher, Tim Colles

Aug 8, 2006

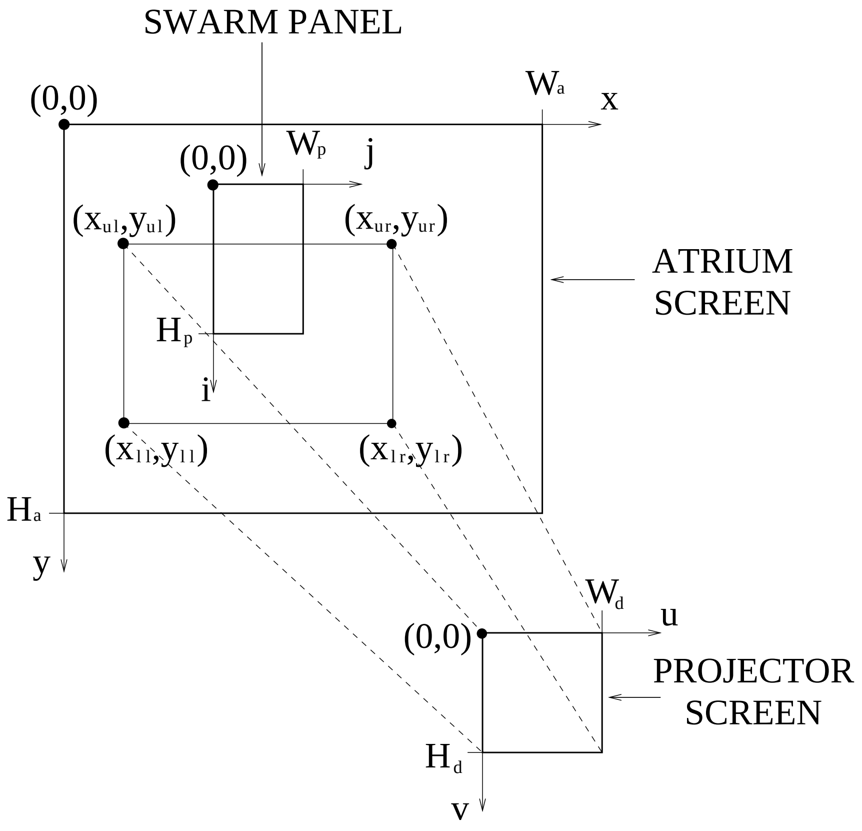
This document is a specification of the software for controlling the Atrium Swarm, and for the control programme.

For the moment, we assume:

1. Panels move with constant speed during a specified segment, but may have a different speed in different segments.
2. Panels receive video data at a constant rate during a specified segment, but may have a different rate in different segments.
3. All panels can move and receive data simultaneously.
4. If a panel has no active programme at a given time, then it stays where it is.
5. There are P panels.
6. The projection geometry is independent of the content and panels.
7. There may be more than one projector, and they might overlap to increase brightness.
8. The panels are position controlled, specified by desired (x, y) points.

1 Swarm Geometry

The general atrium projection geometry is shown here:



The atrium display area is a physical rectangular area H_a mm high by W_a mm wide. Mapped onto this are notional wall coordinates defined with origin at the upper left. Atrium coordinates lie in the ranges $h \in [0, 10000], w \in [0, 10000]$, which covers the whole physical space, at approximately mm resolution. Coordinates are real numbers. (Conversion factors $\alpha_h = H_a/10000, \alpha_w = W_a/10000$ of mm per coordinate value are measured.)

Panels are also rectangular areas H_p mm high by W_p mm wide. Each panel also has panel coordinates defined with origin at the upper left. Coordinates are real numbers. Panel coordinates lie in the ranges $h \in [0, 1000], w \in [0, 1000]$, which covers the whole panel, at approximately mm resolution. (Conversion factors $\beta_h = H_p/1000, \beta_w = W_p/1000$ of mm per coordinate value are measured.)

Each data projector has a rectangular area H_d pixels high by W_d pixels wide. Each projector also has display coordinates defined with origin at the upper left. Coordinates are real numbers. Display coordinates lie in the ranges $h \in [0, 1000], w \in [0, 1000]$, which covers the whole display, at approximately 1 pixel per display coordinate resolution. (Conversion factors $\gamma_h = H_p/1000, \gamma_w = W_p/1000$ of pixel per coordinate are measured.)

The display projects its contents to a rectangular region on the atrium display area with the following coordinates:

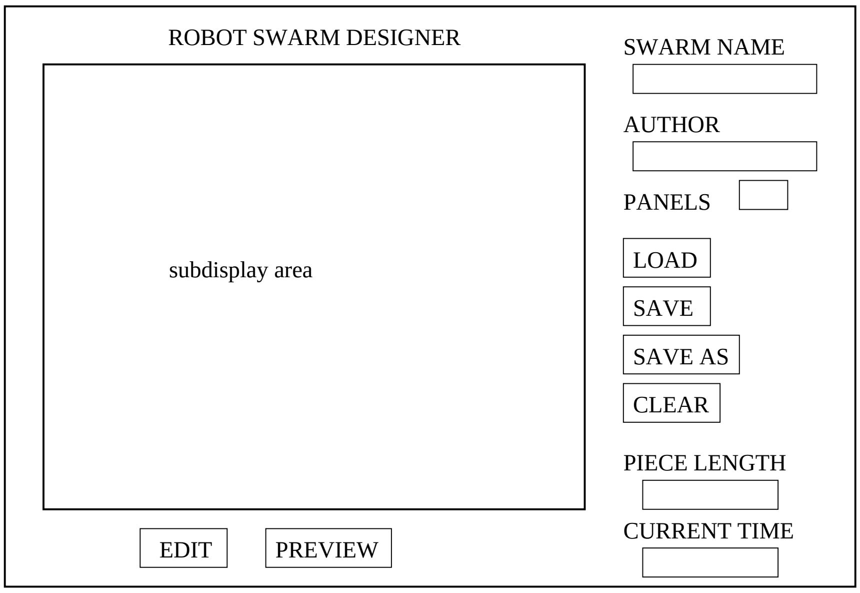
Display coordinate position	Display pixel position	Atrium coordinate position	Atrium physical position
(0,0)	(0,0)	(x_{ul}, y_{ul})	$(\alpha_w x_{ul}, \alpha_h y_{ul})$
(0,1000)	$(0, 1000\gamma_h)$	(x_{ur}, y_{ur})	$(\alpha_w x_{ur}, \alpha_h y_{ur})$
(1000,0)	$(1000\gamma_w, 0)$	(x_{ll}, y_{ll})	$(\alpha_w x_{ll}, \alpha_h y_{ll})$
(1000,1000)	$(1000\gamma_w, 1000\gamma_h)$	(x_{lr}, y_{lr})	$(\alpha_w x_{lr}, \alpha_h y_{lr})$

2 Swarm Data Model

A programme is a hierarchical object. At the top level, there is the piece, which consists of instructions for each of the P panels. The instruction for a panel is a trajectory along which the panel moves, while simultaneously receiving projected images from the data projectors. A trajectory is a concatenated sequence of groups, which themselves are decomposed into a concatenated sequence of segments. (The group intermediate allows repositioning and reuse of a set of segments.) A segment is a 4 point spline curve. A different set of image data can be projected onto a panel in each segment.

3 Swarm Design Tool

A JAVA based tool allows artists and engineers to programme where the panels of the swarm will be at a given time and what will be displayed on the panel at that time. The design tool will also allow the resulting programme to be previewed. The top level layout is:



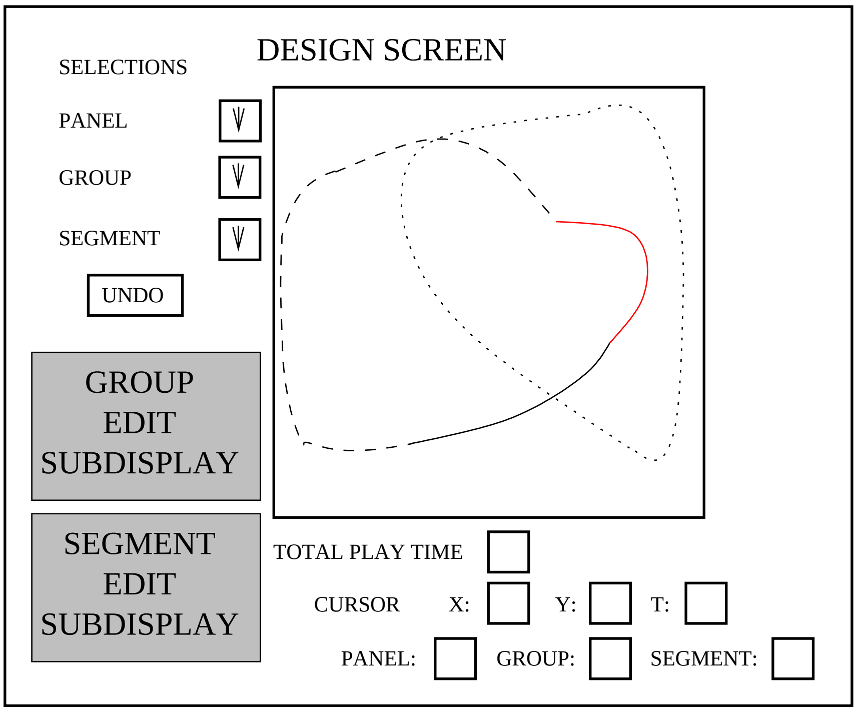
The left area is a subdisplay used for either designing the panel placement and display (see Section 3.1) or previewing a given placement (see Section 3.2). The particular subdisplay is selected by the mutually exclusive highlighted “Edit” or “Preview” buttons.

The right area has several boxes or buttons:

1. **Name** (text box): the title of the piece. Loadable from saved files, but may be entered or overwritten.
2. **Author** (text box): the author of the piece. Loadable from saved files, but may be entered or overwritten.
3. **Load** (button): pops up a panel for entering a file name (default extension .atr) from where the programme is to be loaded.
4. **Save As** (button): pops up a panel for entering a file name (default extension .atr) where the programme is to be stored.
5. **Save** (button): saves the programme in the most recently used file name.
6. **Clear** (button): erases the current program.
7. **Panels** (integer box): the number of panels programmed in this piece. Loadable from saved files, but may be entered or overwritten.
8. **Piece Length** (number box): readonly statistic on the piece’s maximum timecode (seconds).
9. **Current time** (number box): readonly statistic on the current timecode value.

3.1 Panel Placement Subdisplay

This subdisplay allows the creation and manipulation of the panel trajectories and contents. The presented background is a photo of the atrium display area. Work in this panel uses text entry, button selection and mouse drawing. The subdisplay layout is:



A piece consists of 1 or more moving panels. Each panel moves and displays according to its programme, independently of the other panels. If there is some interaction between panels, then this is the result of the designer causing them to move to specified locations synchronously. Each panel moves through a set of one or more segments. The path moved during a segment is a cubic spline, defined by 4 control points (which can be dragged to create different shaped paths. There is also a stationary segment, during which a panel does not move. The designer can assign a video source to each segment, which may come from a previously developed video file, or may be live video from a specified source.

The display area on the right shows the trajectories of the centre positions of each panel. Non-selected panels have dotted trajectories. Selected panels have dashed trajectories, except for the currently selected group, which is solid black. Within the selected group, a segment can be selected, which is shown as a solid red line.

When the cursor is over the display, information is displayed at the bottom right, showing the panel number, group number, segment number, (x, y) coordinate and time associated with the point closest to the cursor.

When a mouse button is clicked, the following actions happen:

1. **Left**: The currently selected segment's closest vertex translates while the mouse is held down.

2. **Middle**: The currently selected segment translates while the mouse is held down.

3. **Right**: The currently selected group translates while the mouse is held down.

4. **CTRL+Right**: The currently selected panel translates while the mouse is held down.

The left area has several boxes or buttons that come in three groups, that reflect the structured nature of a piece. The first group is the "Selections" group.

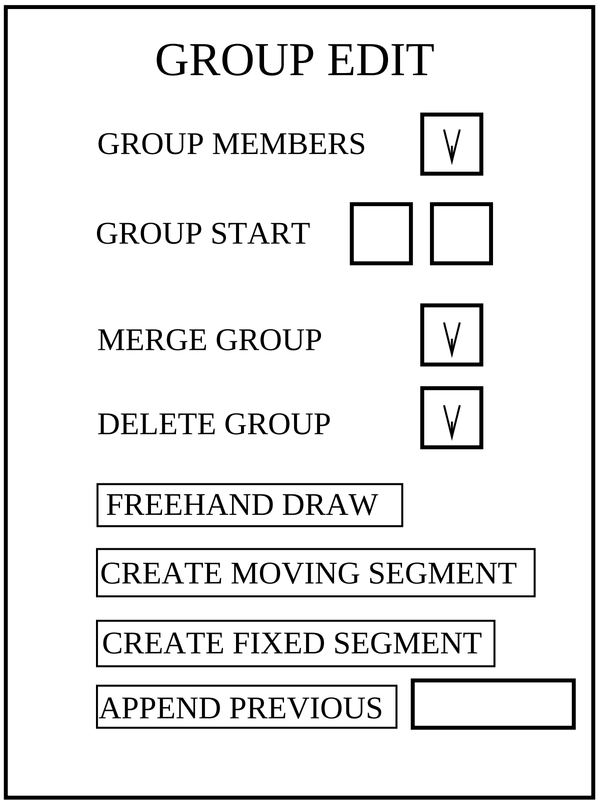
1. **Panel Select** (pulldown menu 1 to P): panel programme being developed.

2. **Group Select** (pulldown menu 1 to N): group of segments to be edited. This should be one of the groups for the current panel (highlighted).

3. **Segment Select** (pulldown menu 1 to N): individual segment to be edited. This should be one of the segments in the current group (highlighted).

4. **Undo** (button): The programme has an internal history list of the editing commands since the last 'save' command. When 'undoing', the last list element is removed and the remaining action commands are re-executed in order.

The second subdisplay is the "Group Manipulation" subpanel, seen here:



When a group is selected, this subdisplay appears and you can:

1. **Group Members** (pulldown menu 1 to N): This is a multiple click pulldown menu for selecting or deselecting segments into the current group.

2. **Group Start** (1 pair of editable control points): The control point coordinates of the group's first point are shown. They can be changed, which moves the whole group. They are also updated if the group is moved by a drag and drop action.

3. **Merge Group** (pulldown menu 1 to N): This merges the segments of the selected group into the current group, deletes the selected group and renumbers the groups.

4. **Delete Group** (pulldown menu 1 to N): This deletes the segments of the selected group and renumbers the segments and groups.

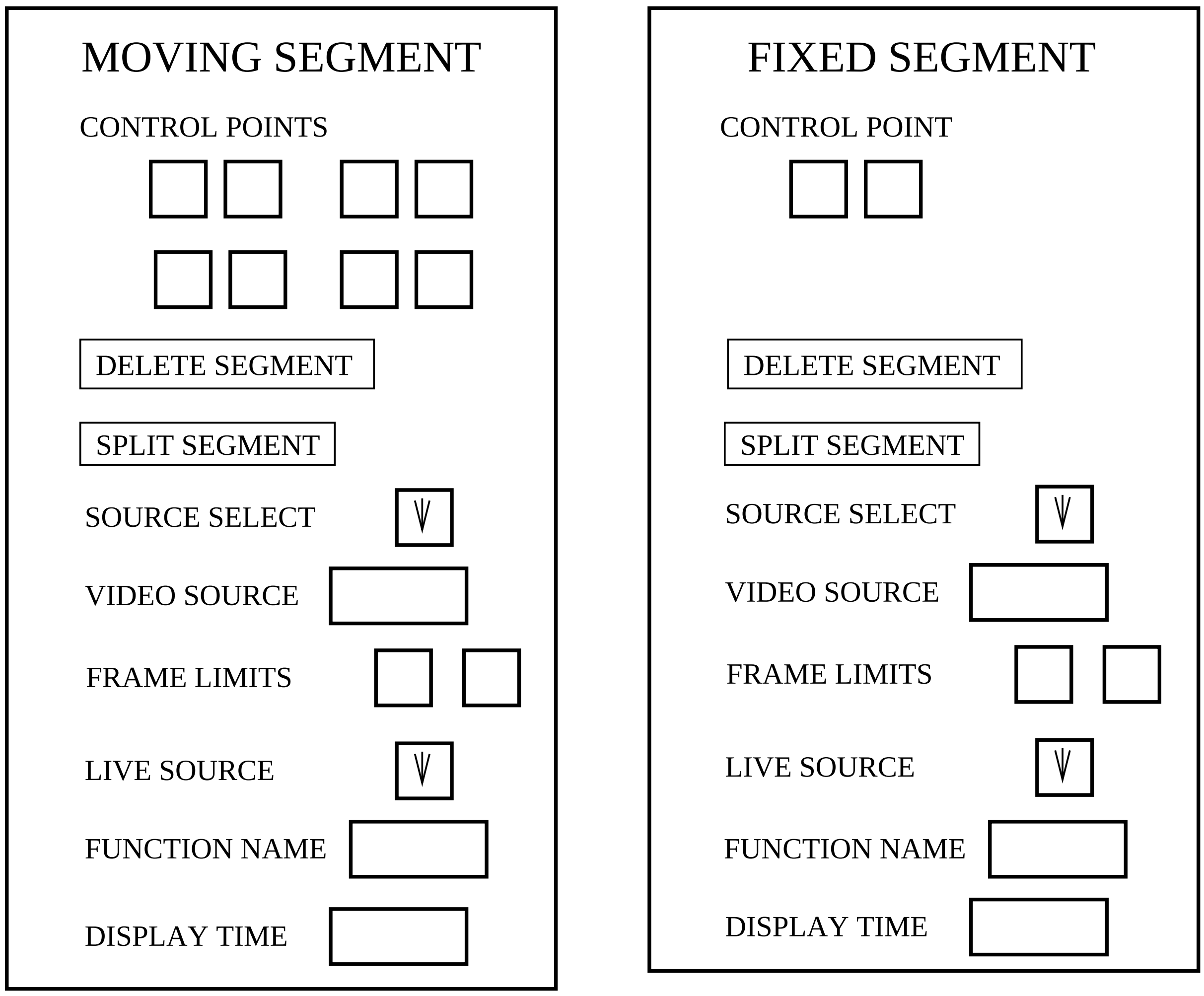
5. **Freehand Draw** (button): The button creates a new group. The user creates the curve by clicking and drawing the left mouse on the display panel. When the mouse is released, the curve is complete. Segments are periodically and automatically created along the curve, which can then be edited.

6. **Create Moving Segment** (button): a new motion segment in a default position is created, added to the group and displayed. The segment is selected. The moving segment manipulation subdisplay is activated (see below).

7. **Create Fixed Segment** (button): a new fixed segment in a default position is created, added to the group and displayed. The segment is selected. The fixed segment manipulation subdisplay is activated (see below).

8. **Append Previous** (button) and box (writable text): loads the segments from the named file and appends them to the end of the current panel's path. The segment translations are adjusted so the segments are connected to the end.

The third and fourth subdisplays are the moving and fixed segment manipulation subpanels, seen here:



These only affect the currently selected segment. The “Moving Segment” subpanel allows:

1. **Control Points** (4 pairs of editable control points): each control point coordinates is shown. The first point affects the overall position of the whole segment. The other three are offsets relative to the first point. These can be set by the author, or are updated during mouse based operations.

2. **Delete Segment** (button): the current segment is deleted, the remaining ones renumbered and the display redrawn.

3. **Split Segment** (button): the current segment is split into two, with a new segment created from the second half of the curve. New control point values are estimated. The display is redrawn.

4. **Source Select**: (pull down menu): showing Prerecorded/Live/Function. Live video

selects the current frame from the specified live source. Prerecorded extracts the required frame from the specified prerecorded source. The function mode calls the specified stand-alone function when a new frame and panel position is needed.

5. **Video Source**: (editable text box): listing file name (in current directory) of file for display.

6. **Frame Limits** (2 writable number boxes): the initial and final frame numbers from the prerecorded source for the display in this segment. If 0, then nothing is displayed. If equal, then only 1 frame is displayed the whole time.

7. **Live Source**: (pull down menu): listing camera ports

8. **Function Source**: (writable text): Name of the function generating panel position coordinates. When executed, a file called 'tmpinput.txt' is written out by the controlled, with contents as below. The file gives the current time and the time to generate positions for (next time), with the assumption of linear movement between specified points. The maximum future time will be a function of the maximum panel speed, the time it takes to execute this function and the maximum position that the panels are to be moved to. More details on this will need to be worked out.

   ```
   current_time next_time number_of_panels
   x1 y1        % panel 1 position
   x2 y2        % panel 2 position
   x3 y3        % panel 3 position
   ...
   xp yp        % panel p position
   ```

   After the function executes, it writes its results to a file called 'tmpoutput.txt', which is read in by the controller for positioning the panels and displaying their contents. The format of the file is:

   ```
   x1 y1 tmpimg1.ppm       % panel 1 position & displayed image
   x2 y2 tmpimg2.ppm       % panel 2 position & displayed image
   x3 y3 tmpimg3.ppm       % panel 3 position & displayed image
   ...
   xp yp tmpimgp.ppm       % panel p position & displayed image
   ```

9. **Display Time** (writable number box): the time length in seconds that the video stream is to be displayed while moving through this segment. Motion is assumed to have constant velocity and display rate is constant.

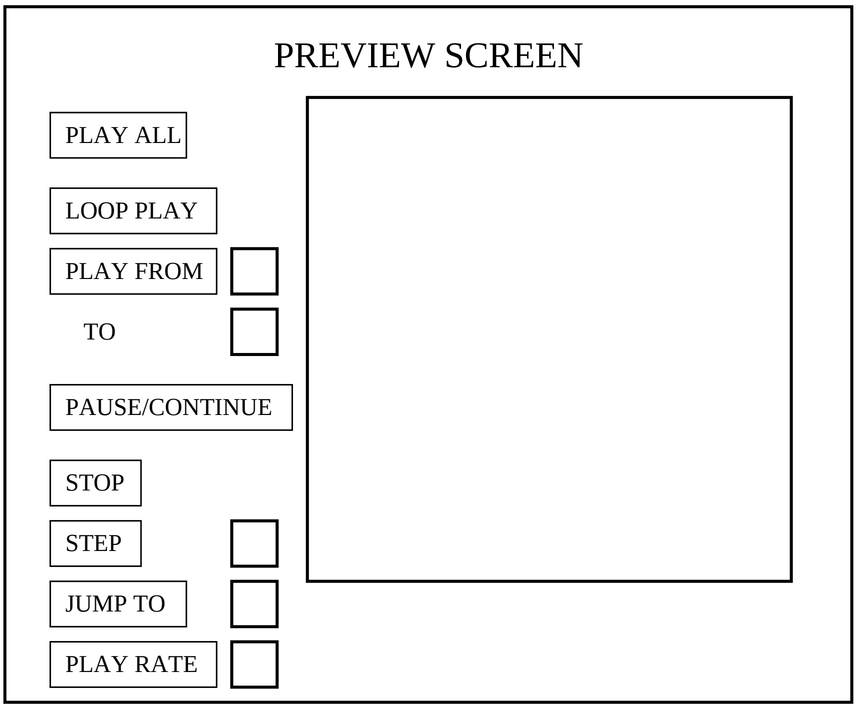
The fourth subdisplay is the "Fixed Segment" group and overlaps the "Moving Segment" display. It is identical to the "Moving Segment" display except for the control point:

1. **Control Point** (1 pairs of editable control points): the control point coordinates are shown. They can be changed by the author.

3.2 Preview Subdisplay

This subdisplay allows the designer to preview the results of the design, as a whole, subsequence or individual frame. The display shows a presentation of the piece, as if it were actually being projected and the panels were in their specified positions.

The subdisplay layout is:



The buttons and display are described below.

The left subarea has several boxes or buttons:

1. **Play All** (button): plays the whole piece.
2. **Loop Play** (button): plays the whole piece repeatedly
3. **Play From/To** (button and 2 number boxes): plays the piece from the first timecode to the second timecode. Useful for debugging a piece.
4. **Pause/Continue** (toggle button): pauses/continues the piece
5. **Stop** (button): stops the piece at the current time.
6. **Step** (button and number box): increments/decrements the current timecode by the specified amount and advances/reverses the presentation. Can be used repeatedly to slowly step through the piece.
7. **Jump To** (button and number box): sets a new timecode and updates the display.
8. **Play Rate** (number box): sets the display to show X performance seconds per clock second (for faster/slower presentation).

The right subarea shows the current rendered view of the piece. The presented background is a photo of the atrium display area. A faint dashed box shows the limits of each data projector's display. Overlaying the background is a faint (0.2 ?) intensity image of the current projector full screen content. Overlaying this are faint solid boundaries of the P panels rendered furthest to closest. Finally, the portion of the data projector output that hits the screens is rendered at full intensity. A typical view is seen in the figure above.

4 Swarm Programme Executer

This programme sends commands to the mechanical panel controllers, and imagery to the data projectors based on. It also acquires visual feedback from the atrium to initially calibrate the swarm, and potentially to obtain feedback as the panels move.

At the moment, we do not have enough information about the mechanical control of the swarm, so here we list a draft set of requirements on the executer.

1. The executer will take as inputs the XML swarm programme and any referenced image data sources (including live data). Presumably the motor controllers will supply current position information also used as control input.

2. The executer will probably be a C/C++ programme.

3. Its outputs are commands (ascii text strings?) to the mechanical controller(s) and video frames to the data projector(s).

4. The controller will check positioning commands for correctness, including making sure that the panels are not commanded to move beyond the limits of the physical display space (minus some tolerance distance).

5. The controller might have visual input to track (using IR LEDs?) the panels to improve positional accuracy and timing. The first version is planned to not have visual feedback.

6. At each time interval, each panel is required to be at a given location and each display projector is required to project a given image.

7. The executor will convert display position coordinates into physical control information (possibly actuator coordinates or speeds).

8. Calibration will link the actuator coordinates to the display coordinates.

9. The executor will need to run in real-time, with some sort of natural clock. Possibly this will be 25 cycles per second, with a new video frame displayed at each cycle, and the robot position updated. An initial estimate of top speed is 1000 mm/sec and thus about 10 sec to cross the atrium, suggesting 40 mm/video frame. This seems too jerky, so faster control will be required. How fast is possible depends on the position control mechanism and how fast is the communication channel.

10. The executor will need to take account of how many data projectors there are and their placement. There may be multiple projectors covering a given position (in order to increase brightness) and so multiple projectors might project the same content. More than one panel might appear in the projection field for a projector, so multiple input sources might need to be projected. Adjacent projection fields might overlap slightly so calibration might be needed to prevent overlighting.

5 Swarm Programme Record

This section describes the XML data file that encodes a Swarm Programme. The strings TTTT denote text strings and NNNN denote numbers.

```
<swarmprogramme name="TTTT" designer="TTTT" date="TTTT">

  <!-- panellist lists the panels -->
  <panellist number_of_panels="NNNN">

    <!-- each panel with "id" displays a number of groups of segments -->
    <panel id="NNNN" number_of_groups="NNNN">

      <!-- this group with id groupid is translated to (pointx,pointy) -->
      <!-- the first group is at (0,0) and the rest should be head-to-tail -->
      <panelgroup groupid="NNNN" pointx="NNNN" pointy="NNNN">
      ... more panel groups
    </panel>
    ... more panels
  </panellist>

  <!-- this is the list of all groups in the programme -->
  <grouplist number_of_groups="NNNN">

    <!-- the set of segments in a group, each translated to (pointx,pointy) -->
    <!-- the first segment is at (0,0) and the rest should be head-to-tail -->
    <group groupid="NNNN" number_of_segments="NNNN">
      <groupseg segid="NNNN" pointx="NNNN" pointy="NNNN">
      ... more group segments
    </group>
  </grouplist>

  <!-- this is the list of all segments in the programme -->
  <segmentlist number_of_segments="NNNN">

    <!-- a segment displays for displaytime and is of type=VIDEO/LIVE/PROG -->
    <!-- it has motion=MOVING/FIXED -->
    <segment id="NNNN" displaytime="NNNN" segmenttype="TTTT" motion="TTTT">
      <videostream filename="TTTT" startframe="NNNN" endframe="NNNN">
      <livestream sourcename="TTTT">
      <progstream progname="TTTT">

      <!-- spline control points have ids=0/1/2/3. Point 0 is at (0,0) -->
      <controlpoint id="NNNN" pointx="NNNN" pointy="NNNN">
      ... 0 or 3 more points
    </segment>
    ... more segments
  </segmentlist>

</swarmprogramme>
```

6 Development Time

We estimate

- **Design Tool**: 3-4 person months, mostly at the beginning
- **Display Controller**: 3 person months, 2 months in parallel with or after the Design Tool and an additional month after physical installation of the equipment.