Final Project Report

COMP 471

Real-time Video Processing

I- Sight- Painting

Concordia University

Monday, December 11, 2006

Final Project Report
COMP471

Group Members

Sami Al-Khudri	ID- 5231256
Aliaa Shubbar	ID- 4628977

Project Name & URL

I-Sight-Painting

Project URL- http://sami.alkhudri.com

Project Description

The inspiration behind our project was based on the I/O brush created by Ryokai, K., Marti, S., Ishii, at the MIT Media Lab.

“I/O Brush is a new drawing tool to explore colors, textures, and movements found in everyday materials by "picking up" and drawing with them. I/O Brush looks like a regular physical paintbrush but has a small video camera with lights and touch sensors embedded inside. Outside of the drawing canvas, the brush can pick up color, texture, and movement of a brushed surface. On the canvas, artists can draw with the special ‘ink’ they just picked up from their immediate environment. “ [web.media.mit.edu]

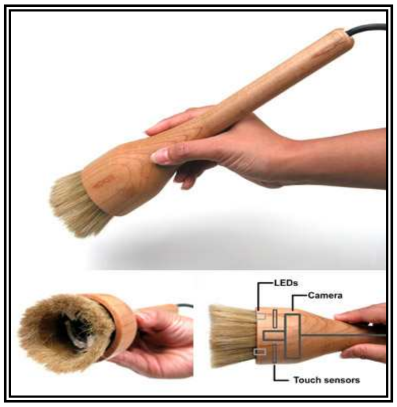


Figure 1 I/O Brush

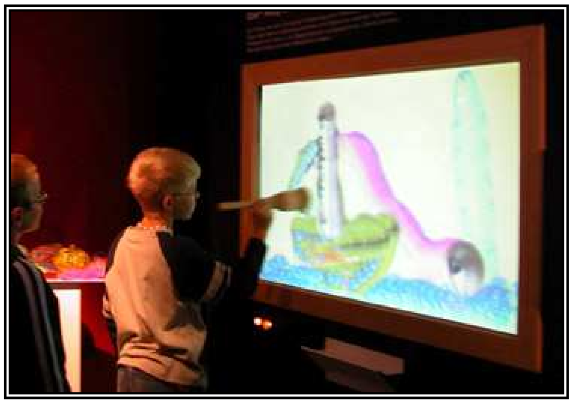


Figure 2. I/O Brush in action

Our Goal was to create a new type of interaction that eliminates the used of the MIT I/O brush and requires a touch screen canvas. Our project allows the user to create his or her masterpiece using only live video input and movement detection. This enables the user to “draw” using live video as well as images as their “ink” to and to interact with the canvas by movement.

How I- Sigh Painting works

Two cameras are used in this project. One is used to record a live video image from the immediate environment. This image translates onto the canvas and will act as paint (a picture can be also used). The other is used to track the hand gestures of the user. This is done by wearing a glove that has a color different from the user’s clothes. The patch tracks the glove based on its RGB value. For example a red glove would yield a RGB value close to 1. 0. 0.

As the person moves his or her hand, the recorded image will move accordingly across the screen. We can also use pre-recorded videos and images as the palette used to project onto the path drawn by the person. Using jitter we create a path by tracking the users movement in front of the camera, once the path is created we will fill the path with multiple instances of the video ink. This gives texture, color and movement to the drawing.

Using jitter we will create a path by tracking the users movement in front of the camera by using the jit.findbounds we are able to track the x y coordinates of the user’s movements. Once the path is created we will fill the path with multiple instances of the video ink. This gives texture, color and movement to the drawing.

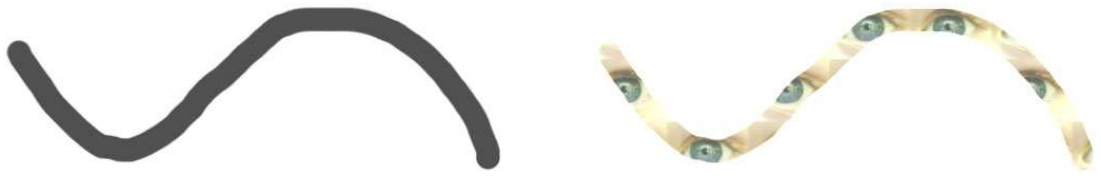


Figure 3. Path created by tracking user’s movement, and path with Eye Ink (video of an eye) applied

Technical Aspect

Our entire project was created using Jitter. To get a better understanding of the patch used refer to the figure 4.

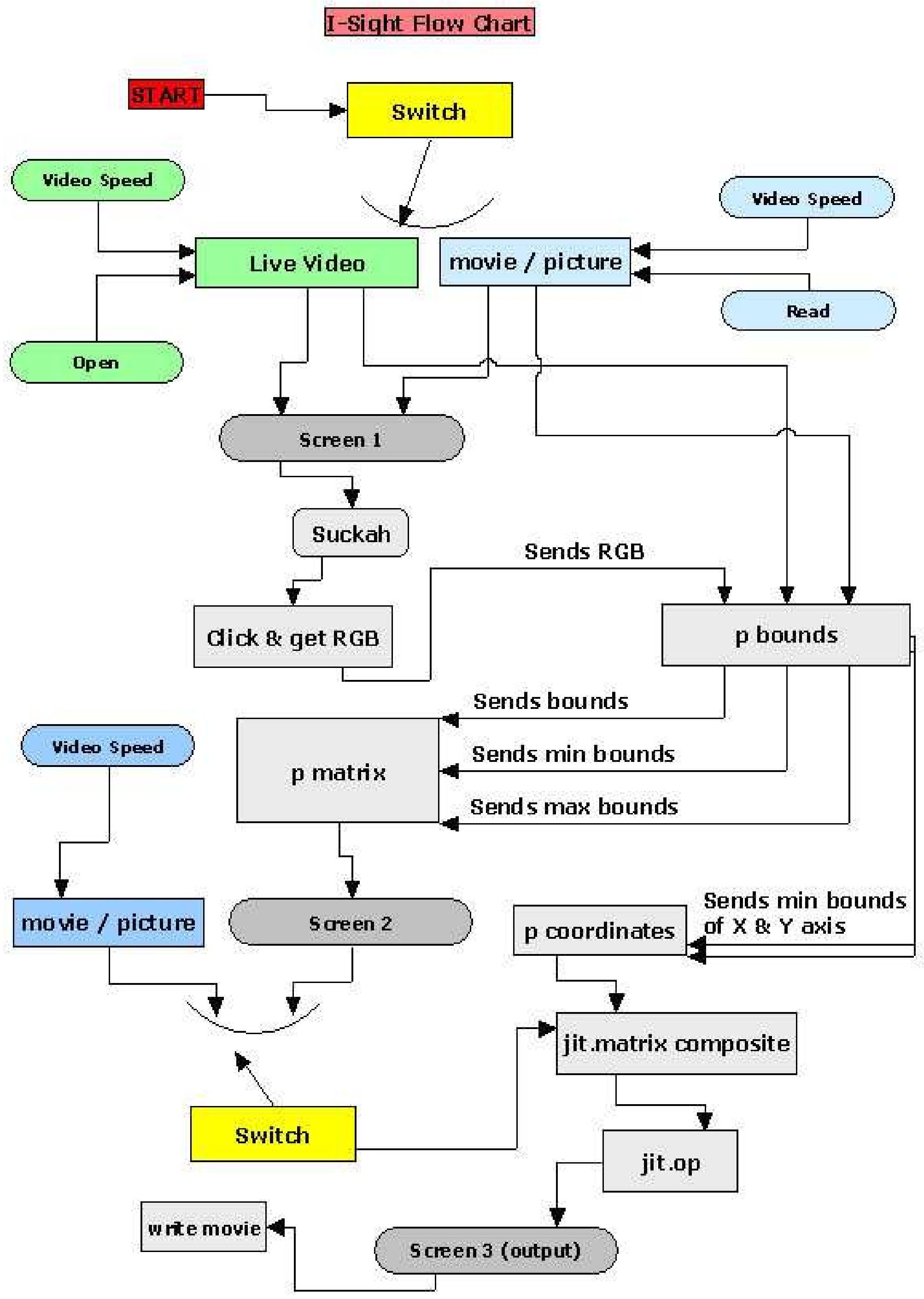


Figure 4. Flow Chart of I-Sight-Painting Patch

Mathematics & techniques used in the project:

1) Color Tracking: We used the “suckah” object on the live video screen. We put "suckah" onto the live video tracker to get the RGB color beneath, or feed in any screen coordinates to get the RGB values of that pixel. The suckah object reports the RGB values of any pixel on the screen (the video window) that the suckah object overlays. Then when the user clicks onto the desired color to be tracked the RGB will be sent to a sub patch called "bounds" that increases the range of the color by adding 0.05 to the maximum value and subtracting 0.05 from the minimum value. With this technique we can guaranteed to track the colour desired despite lighting conditions.

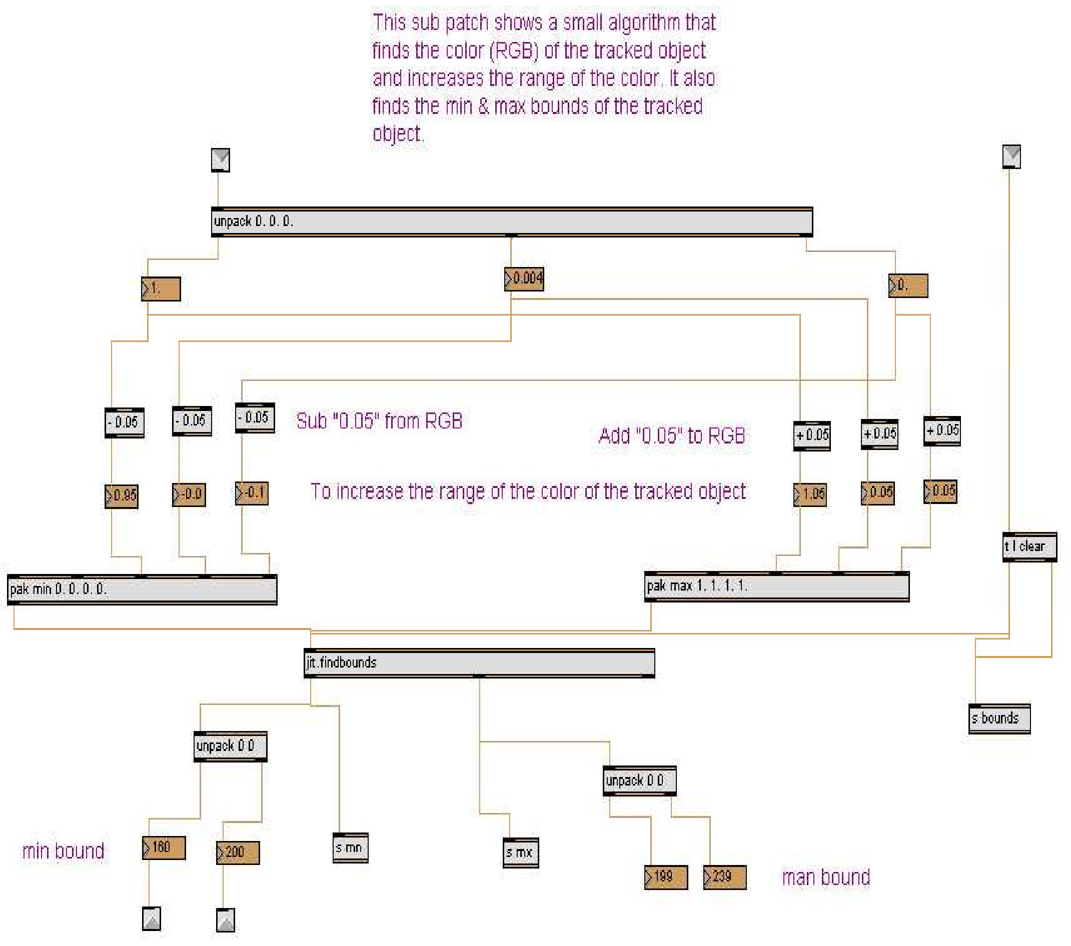


Figure 5. Sub patch that finds the RGB

Using colour encoding we can define an image is an enormous two-dimensional array of color values, pixels, representing the three primary colors. This allows the image to contain a total of 256x256x256 = 16.8 million different colors. The resulting image is coded in a two dimensional spatial domain. We use vector representation of colour.

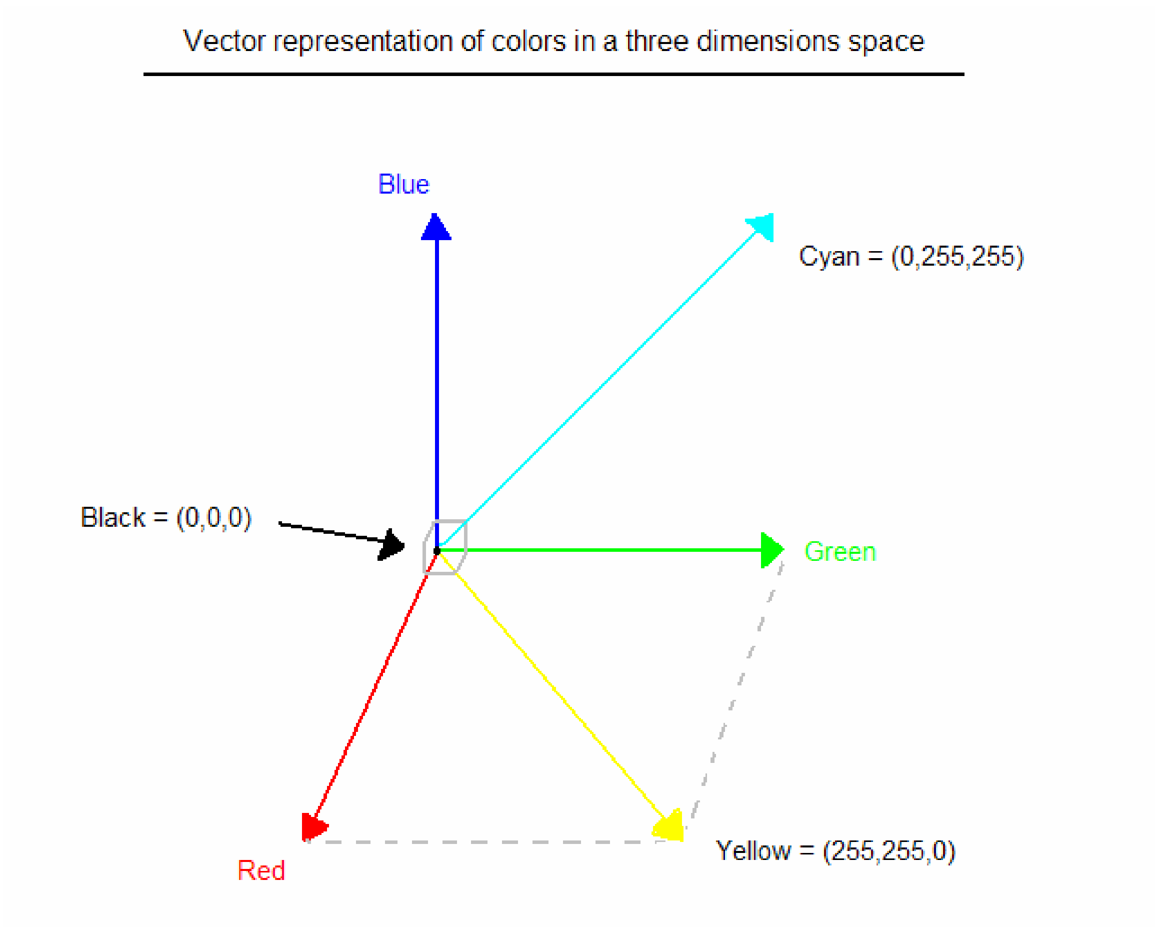


Figure 6. Vector representation of colour

We can quantify the 'difference' between two colors by computing the geometric distance between the vectors representing those two colors. Lets consider two colors C1 = (R1,G1,B1) and C2 = (R2,B2,G2), the distance between the two colors is given by the formula :

$$D(C1,C2)=\sqrt{(R1-R2)^2+(G1-G2)^2+(B1-B2)^2}$$

(R1)

Algorithm 1: Edge Detection

```
• For every pixel ( i , j ) on the source bitmap
    o Extract the (R,G ,B) components of this pixel, its right
      neighbour (R1,G1,B1), and its bottom neighbour (R2,G2,B2)
    o Compute D(C,C1) and D(C,C2) using (R1)
    o If D(C,C1) OR D(C,C2) superior to a parameter K, then we
      have an edge pixel !
```

In our case we wanted to compare each pixel with a given colour, namely C1.

Algorithm 2: Color extraction

```
• For every pixel ( i , j ) on the source bitmap
    o Extract the C = (R,G ,B) components of this pixel.
    o Compute D(C,C0) using (R1)
    o If D(C,C0) inferior to a parameter K, we found a pixel
      which colour's matches the colour we are looking for. We
      mark it in white. Otherwise we leave it in black on the
      output bitmap.
```

2) Finding the bounds: We find the bounds of the tracked object by using "jit.findbounds". The jit.findbounds object scans a matrix for values in the range [min, max] and sends out the minimum and maximum points that contain values in the range [min, max]. The minimum point is sent as a list out the leftmost outlet, and the maximum point is sent as a list out the second outlet. The object jit.findbounds takes a minimum and a maximum value as attributes and looks through the entire matrix (the video) for values that fall within the range you specify. It then sends out the cell indices that describe the region where it found the designated RGB values. In effect, it sends out the indices of the bounding region within which the values appear.

3) Displaying onto the screen: We display the tracked object onto the screen by using "jit.matrix".

The image matrix can be denoted as follows:

$$I = [I\ (i,j);\ 0 \le i \le N-1,\ 0 \le j \le M-1\ ]$$

Where (i.j) = (row, column)

I (i,j) = image valule at (i,j)

$$\mathbf{I} = \begin{bmatrix} I(0,0) & I(0,1) & \dots & I(0,M-1) \\ I(1,0) & I(1,1) & \dots & I(1,M-1) \\ \vdots & \vdots & & \vdots \\ I(N-1,0) & I(N-1,1) & \dots & I(N-1,M-1) \end{bmatrix}$$

(Class Slides)

- dim int list[32] The dimensions of matrix data (default = 1 1)
- dstdimend int list[32] The destination dimension end position (default = all dim values minus 1)
- dstdimstart int list[32] The source dimension start position (default = all 0)
- srcdimend int list[32] The source dimension end position (default = all dim values minus 1)
- srcdimstart int list[32] The source dimension start position (default = all 0)
- usedstdim int Destdim use flag (default = 0) When the flag is set, the destination dimension's attributes are used when copying an input matrix to an internal matrix.
- usesrcdim int Srcdim use flag (default = 0) When the flag is set, the source dimension's attributes are used when copying an input matrix to an internal matrix.

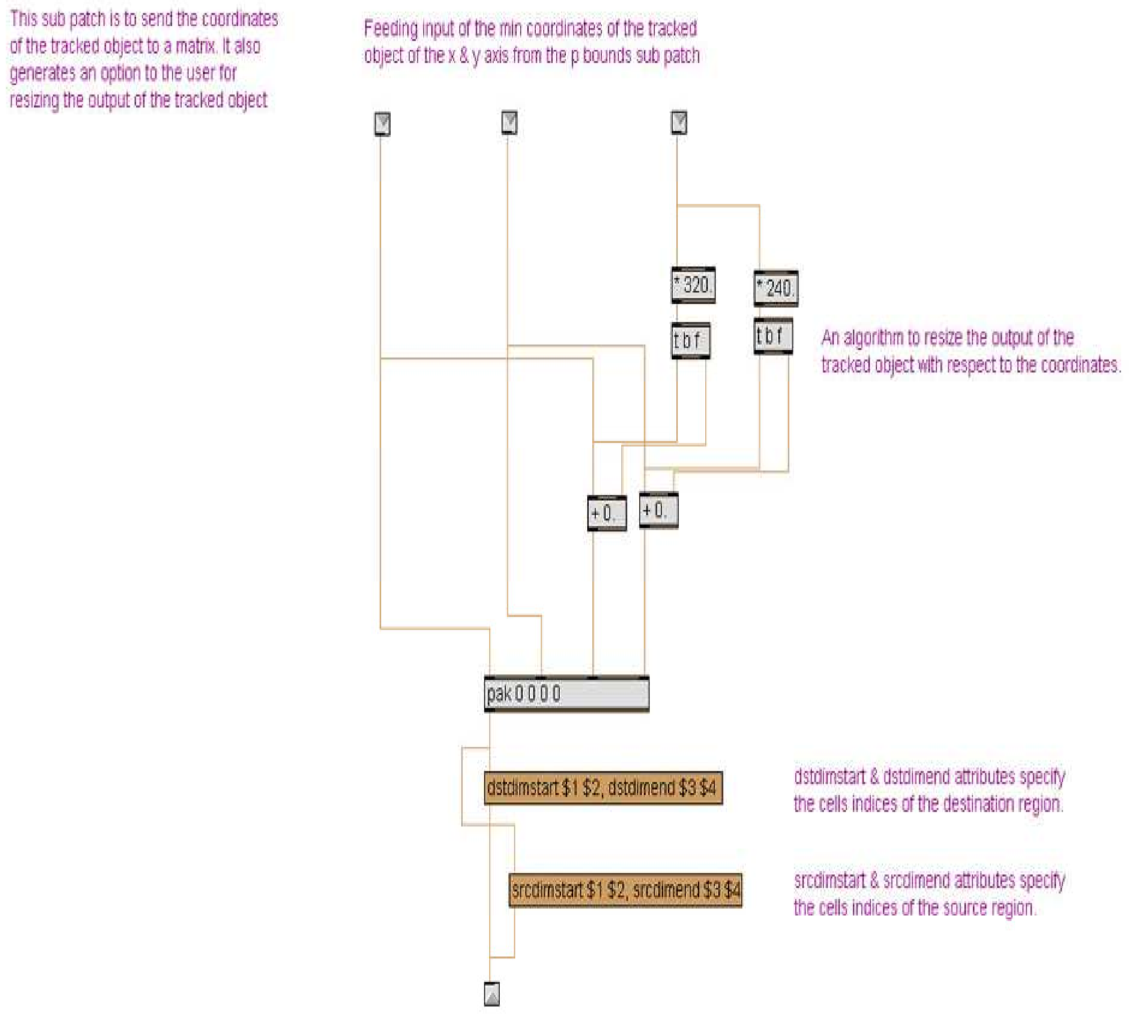

This sub patch is to send the coordinates of the tracked object to a matrix. It also generates an option to the user for resizing the output of the tracked object
Feeding input of the min coordinates of the tracked object of the x & y axis from the p bounds sub patch
* 320.
* 240.
t b f
t b f
An algorithm to resize the output of the tracked object with respect to the coordinates.
+ 0.
+ 0.
pak 0 0 0 0
dstdimstart $1 $2, dstdimend $3 $4
dstdimstart & dstdimend attributes specify the cells indices of the destination region.
srcdimstart $1 $2, srcdimend $3 $4
srcdimstart & srcdimend attributes specify the cells indices of the source region.

Figure7. Sub patch to send minimum & maximum coordinates of the tracked object

4) Continuous & smoothing Path: We have used "jit.op" in order to use its operators especially the MAX operator. The reason for using MAX is to avoid overriding the new frame onto the old frame otherwise the path will be discontinuous at the middle of the screen. By using this technique, the MAX operator compares the new frame with the old frame and retains the cells with the brightest value. Refer to figure 8 and 9.

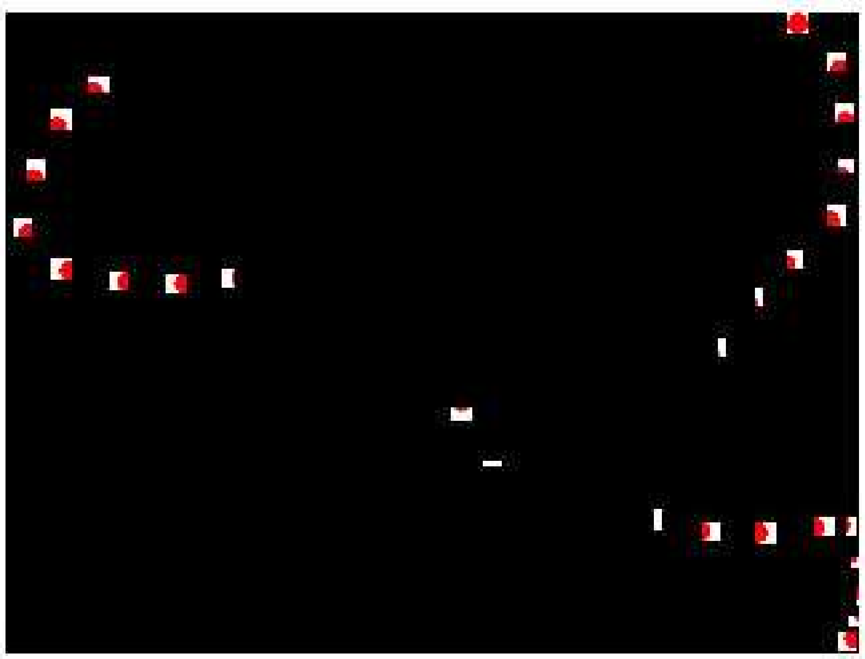


Figure 8. Before implementation of jit.op max

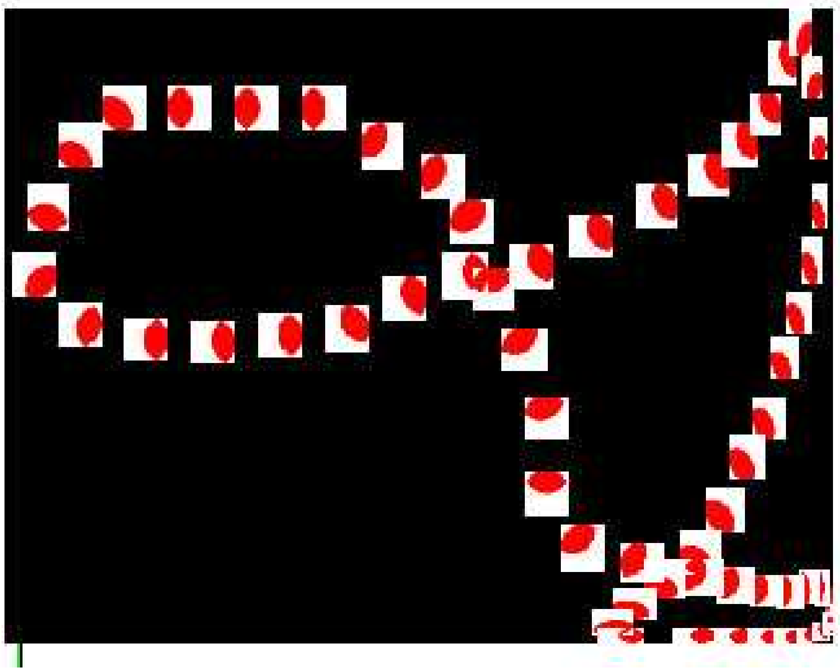


Figure 9. After implementation of jit.op max

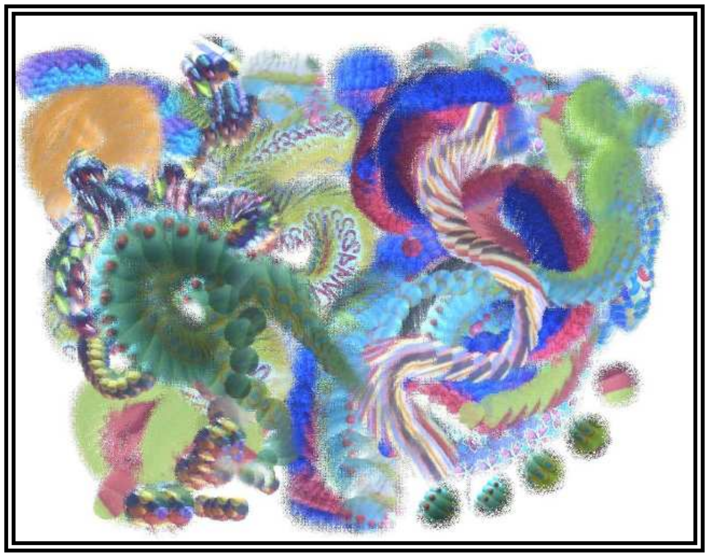
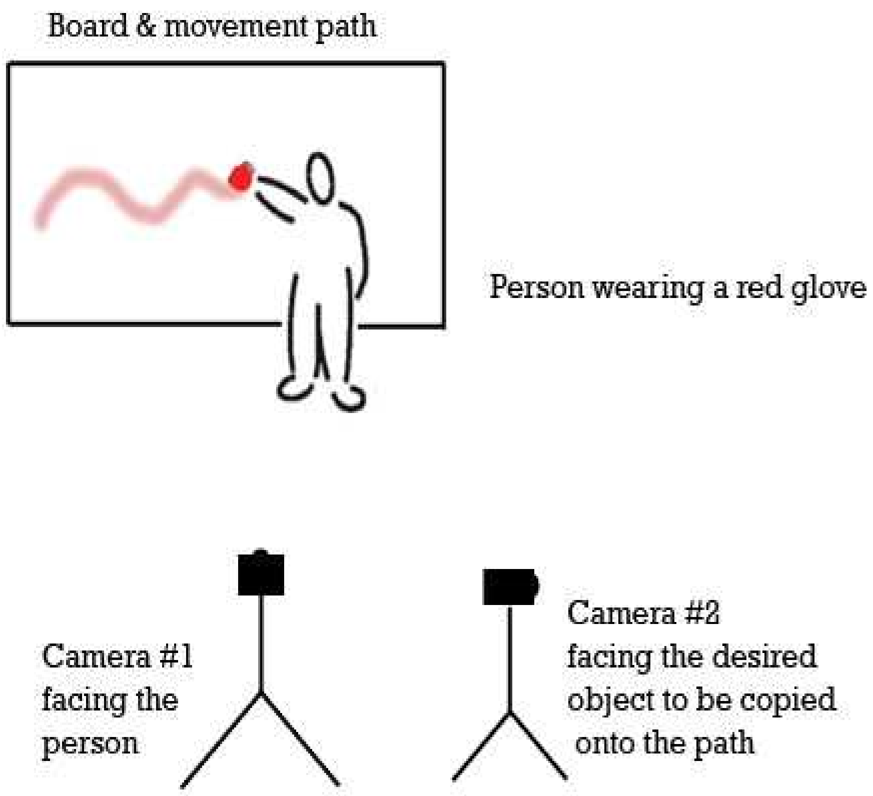


Figure 10. Example drawing

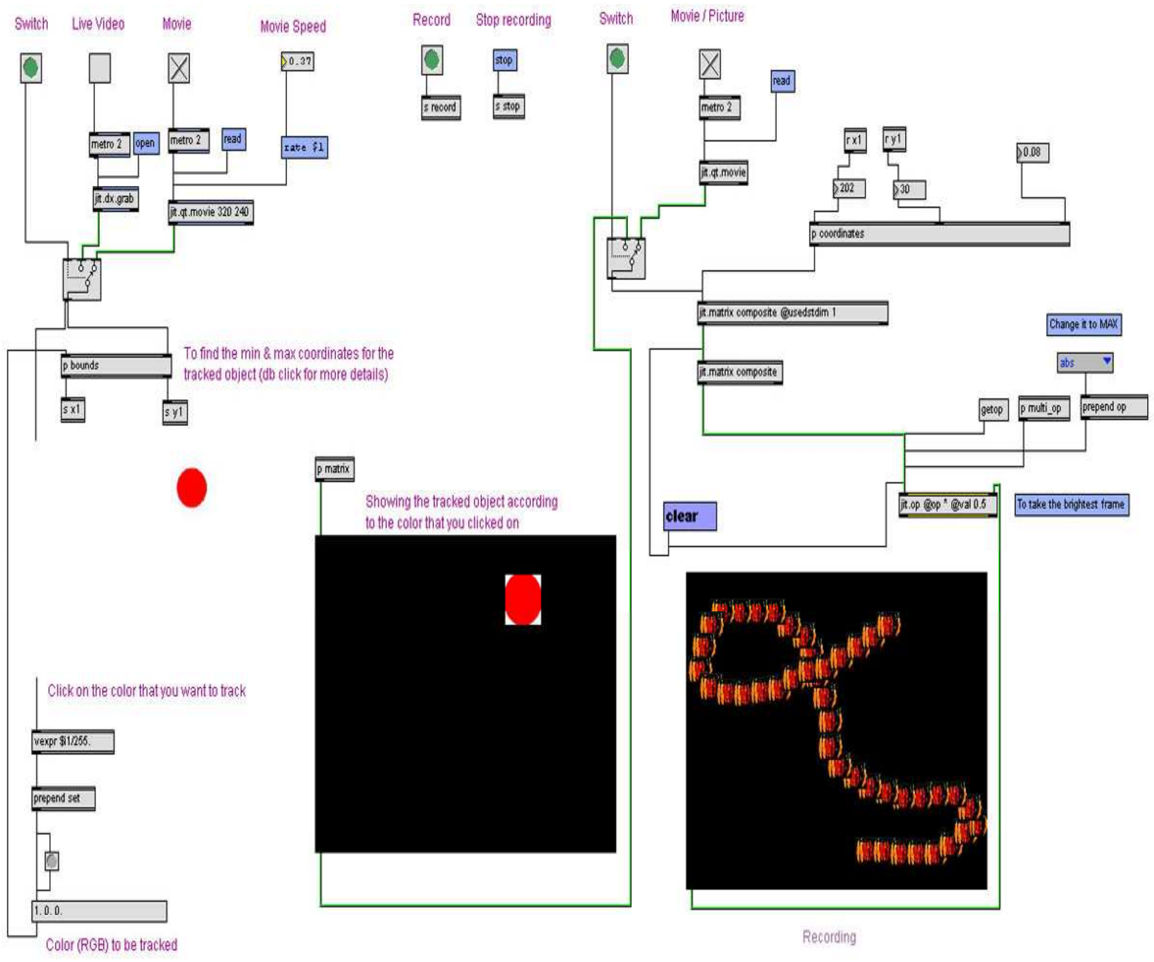
Setup


Board & movement path
Person wearing a red glove
Camera #1 facing the person
Camera #2 facing the desired object to be copied onto the path

Equipment needed

1. Laptop with Max MSP / Jitter
2. Projector
3. I-sight (2)
4. Black Screen

Final Patch



Challenges

One of the biggest challenges of the project is the creation of the path by tracking the users' movement. A second challenge is to properly insert the recorded movies along the path so it will give the impression that the path was painted with the movie. The third and final challenge is to do the overlying at the same time the path is created.

Limitations

Due to time constraints, and the type of interaction with the "canvas" using movement, it will be difficult to integrate pressure detection. Also due to the time constraint for this project some "bugs" might be present in the system, but this will be a good start to improve this technology. I should also mention that we were 4 people at first, but one person has been difficult to reach due to medical purposes. Therefore, we were not able to implement all the technical aspects we aspired to.

People and Roles

Names	Research	Jitter Programming	Documentation	Website	Set Up	Testing
Ahmad Mansur			**X**			
Aliaa Shubbar	Layering Colour Tracking	Layering	**X**		**X**	**X**
Sami Al-Khudri	Colour Tracking Layering	Layering/Integration	**X**	**X**	**X**	**X**

Conclusion

The concept behind this installation is an aesthetic one, as well as a practical one. This installation can be used much like the I/O brush, especially for children's purposes. The interaction is a conscious one where the user is aware of his or her presence in the installation.

REFERENCES

MIT IO Brush
http://web.media.mit.edu/~kimiko/iobrush/
http://web.media.mit.edu/~kimiko/iobrush/images/s_iobrush_aec_02.jpg

Jitter library
http://www.cycling74.com/products/jitter

Video example
http://web.media.mit.edu/~kimiko/iobrush/iobrush_quicktime_small.mov