User-friendly stylus and video surface CAD system

Jeremy Schwartz
Paul Peeling
Faraz Ahmad
Overview

Position detector

Command Handler

Video Display
Position detector

Tasks

• Take a camera image as input
• Filter camera image for green blob
  → This step to be done on the fly!
• At end of each frame, calculate centroid of green blob
• Output stylus position (x, y)
Position detector

Block Diagram

• Only record the points that register as ‘green’
  → Throw everything else away

• Do this on the fly…we must pipeline!
Video input

Global clock

Global reset

Video processing unit (tracks pixel location, outputs luminosity information)

Filter value register (either chromiticity or luminosity)

Asynchronous comparator

Divider enable, # of passed points

Frame done

Video Input FSM

Divider

Accumulator

Output register

Pipeline register

Asynchronous comparator

Register

Pass pulse

Divider enable

Stylus position

Z represents width of information about one pixel
Y represents height of screen (in pixels)
X represents width of screen (in pixels)
State Transition Diagram For Video Input FSM

- **WAIT FOR PASS**
  - Accum_enable = 1
  - Increment # of passed points
  - Frame done = 0

- **ACCUMULATE**
  - Accum_enable = 1
  - Increment # of passed points

- **DIVIDE**
  - Divide enable = 1

- **RESET**
  - # of passed points = 0

- **REGISTER OUTPUT**
  - Output register LE = 1

- **IDLE**

- **Frame start = 0**

- **Frame start = 1**

- **Pass pulse = 1**

- **Pass pulse = 0**

- **Frame done = 0**

- **always**

- **always**
Command Handler

- Screen position and button click input
- Maps the screen position to either a drawing area position, or a menu / toolbar command
- Major/Minor FSM structure controls flow of drawing algorithms
- Output to a frame buffer containing information about the objects
- Frame buffer decoded into RGB
Screen Mapping

Menu

Tools

Drawing area
(x,y)
Command Router

- Stores the current state of a drawing operation (e.g. `got_centre_of_circle`) in the mode register.
- Uses ‘next’ signals to tell the minor FSMs to complete the next operation given the new position information from the input.
- ‘Next’ and ‘busy’ signals are muxed to result in simple command-independent structure.
Example: Line-drawing FSM

Bresenham’s Algorithm
Storage of objects

ZBT SRAM storage: 19 address lines, 36 data lines

Address: \[X\text{ coord}[9:0] \quad Y\text{ coord}[8:0]\]

640x480 resolution frame buffer

Data: object identifiers and information

To video display through decoder

Also stored in a lookup table for easy access to related objects, e.g. the other end of a line
Snapping

- Snapping to grid, other objects etc.
Video Display

The objectives for the Video Display are:

• Display Command Toolbar

• Display Drawing Image

• Be able to superimpose this onto a background
Implementation of Video Display

Video Module

Sync Generator

Pixel Clock (31.5MHz)

Digital Clock Manager (DCM)

System Clock

(31.5MHz)
# Milestones

<table>
<thead>
<tr>
<th>Date</th>
<th>Milestones</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 26</td>
<td>Point drawing (to SRAM)</td>
</tr>
<tr>
<td></td>
<td>Camera input</td>
</tr>
<tr>
<td>April 29</td>
<td>Lines, circles, polygons</td>
</tr>
<tr>
<td></td>
<td>B&amp;W display</td>
</tr>
<tr>
<td>May 2</td>
<td>Snapping</td>
</tr>
<tr>
<td></td>
<td>Command &amp; Video integration</td>
</tr>
<tr>
<td>May 6</td>
<td>Pointer input</td>
</tr>
<tr>
<td></td>
<td>Color display</td>
</tr>
<tr>
<td>May 9</td>
<td>Deletion &amp; Movement</td>
</tr>
<tr>
<td></td>
<td>System integration</td>
</tr>
</tbody>
</table>