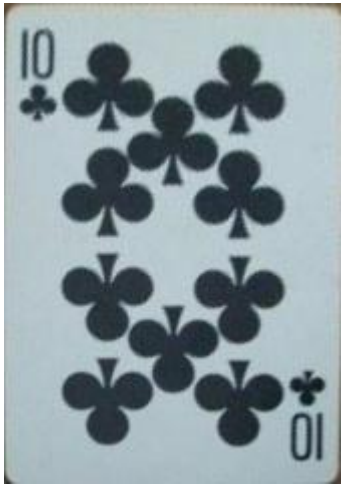
An Image Perspective Correction System

Matt Hollands, Patrick Yang

Problem Statement + Motivation

- Digitization of objects can be done via smartphone camera
- To correctly produce a rectangular object, we must correct for perspective
- We implement a system that solves this problem in hardware.

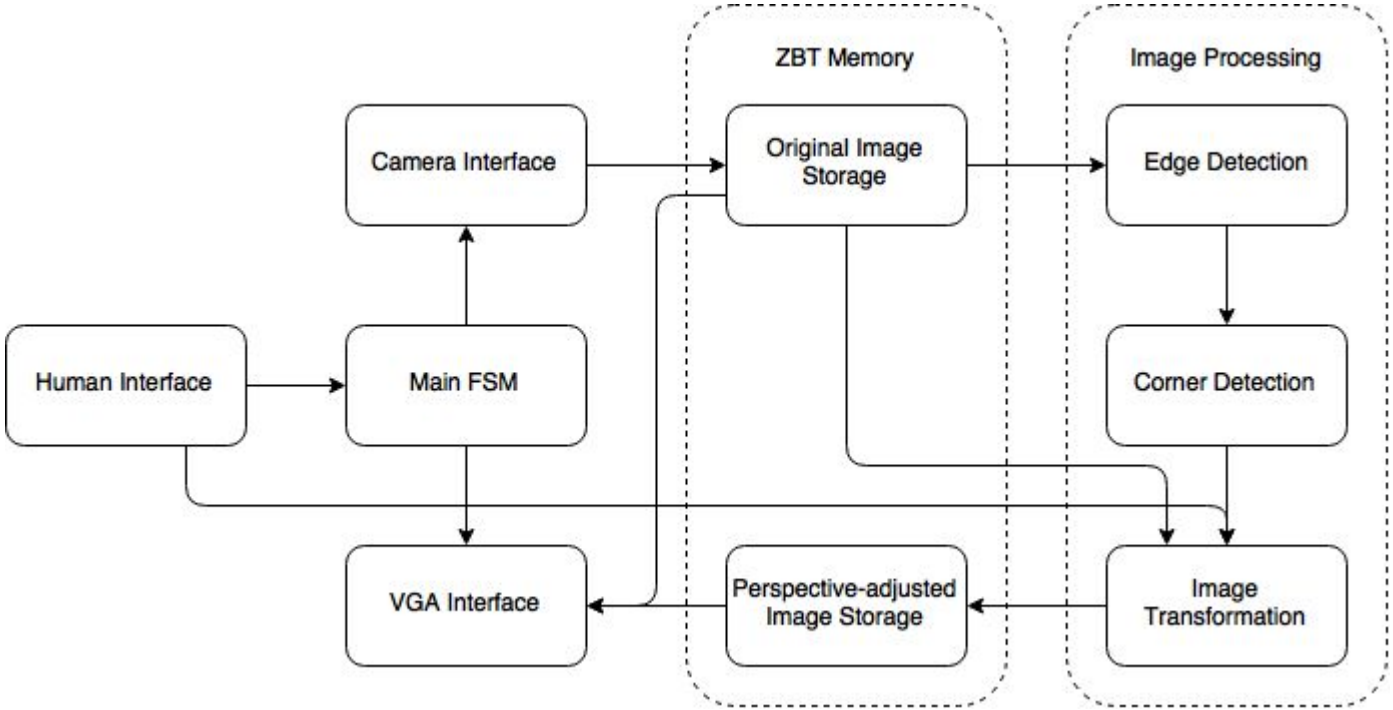


Source:http://opencv-code.com/tutorials/automatic-perspective-correction-for-quadrilateral-objects/

Design Overview

- Store a camera frame in ZBT memory
- Use algorithmic feature recognition techniques to identify the corners of a rectangular object in the camera view, OR let the user specify the corners using a human-interface module
- Compute a transformation that will rectify the perspective
- Save the perspective-corrected rectangle in another bank of ZBT memory

Block Diagram

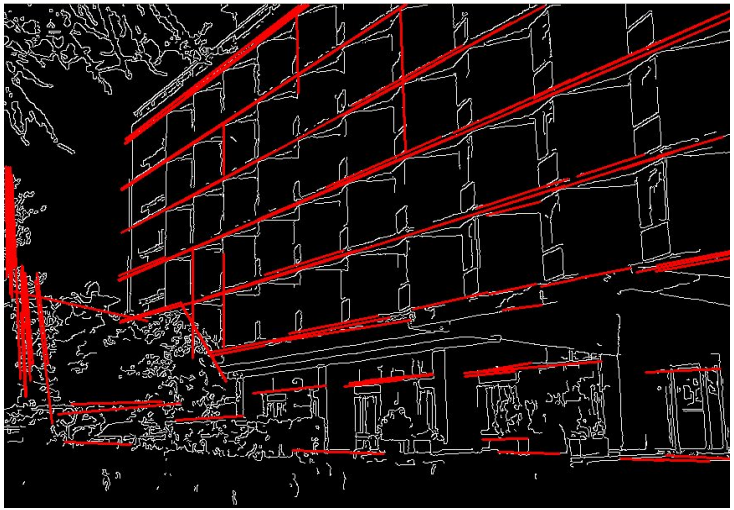


Camera, VGA modules

- Camera module loads single frame into a ZBT memory bank.
- VGA module reads from a ZBT memory bank and displays on the screen along with sprites.
- Must be synchronised with any other module accessing memory.
- The VGA module will be tested by reading an image from memory onto the display.

Corner finding components

- Image processing algorithms required:
  - Canny edge detection for finding edges in the image
  - Hough Transform to find long lines representing the edges on the document



http://docs.opencv.org/2.4.9/modules/imgproc/doc/feature_detection.html

Image transformation components

- A perspective transform is defined by a 9-parameter non-linear map

$$(x, y) \rightarrow \left(\frac{p_1 x + p_2 y + p_3}{p_7 x + p_8 y + p_9}, \frac{p_4 x + p_5 y + p_6}{p_7 x + p_8 y + p_9} \right)$$

- Given four corners of the screen mapping to four corners of the skewed-perspective object, we have 8 equations.

- We can use these to compute the parameters

- Given the parameters and any pixel on screen, we can find the pixel in the original (skewed) image that it maps to

Integration (Main FSM, human interface)

- Hold the state of the overall system.
- Some high level states may include:
  - Camera Viewfinding
  - Image Capture
  - Image Processing/Transformation
  - Processed Image
- Transitions between states will controlled by the user.
- Tested by using test benches to stimulate the inputs

Timeline

- **Week of Nov 2**: Read camera data to ZBT memory, replicate a stored image from ZBT memory to VGA
- **Week of Nov 9/16**: Human interface module, image transformation module
- **Week of Nov 16/23**: Algorithmic edge and corner detection
- **Week of Nov 30**: Integration and end-to-end testing