Mobile Landmark Recognition

Working title - POIRE: Point of Interest Recognition

njgroene, bgroot, ahalma, mtromp, bquiroga

DOAS Progress Meeting

Short project description

- Landmark recognition;
- Computer Vision is done on local device;
- Approximate location is known:
  set of objects is small.

Limited Hardware

We can't use computationally expensive algorithms

- Photos can't be stored in a database, only a symbolic representation;
- Only machine learning algorithms which have a easy to calculate decision boundary.

No lab circumstances

- Different lightning conditions;
- Rotation/translation/scaling;
- Different cameras.

Two sub-problems

- Getting familiar with J2ME;
- Creating a computer vision application wich satifies our demands.

Two major approaches

- Use features which are robust;
- Find a representation of images which is robust.

Possible features

- Skyline detection combined with a descriptor (fft, distance between corners);
- Features of straight lines within an image (orientation, length, color pairs on either side);
- Corner features (location, angle etc.);
- Color histograms;
- Clusters in a colorspace (i.e. HVC).

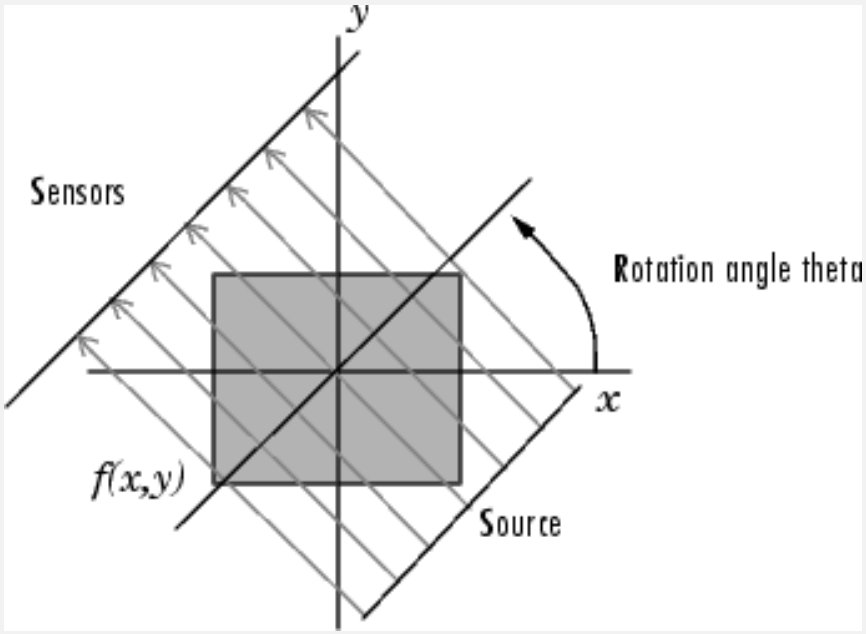
Finding a robust representation

- Problem is hard because of extrinsic variation;
- So: find a way to neutralize this variation;
- Many possibilities...
- Example: Möbius-Radon transformation.

Example: Möbius-Radon transformation

Steps:

- Obtain edge image
- Apply Radon transformation:



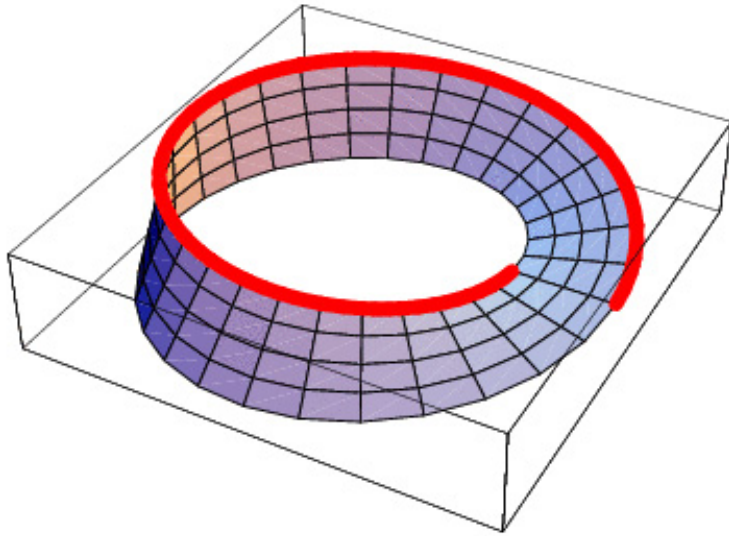
Example: Möbius-Radon transformation

So we have:

Example: Möbius-Radon transformation

- Now connect $\theta = 180$ and $\theta = 0$
- And 'normalize' by shifting important lines to $\theta = 0$.

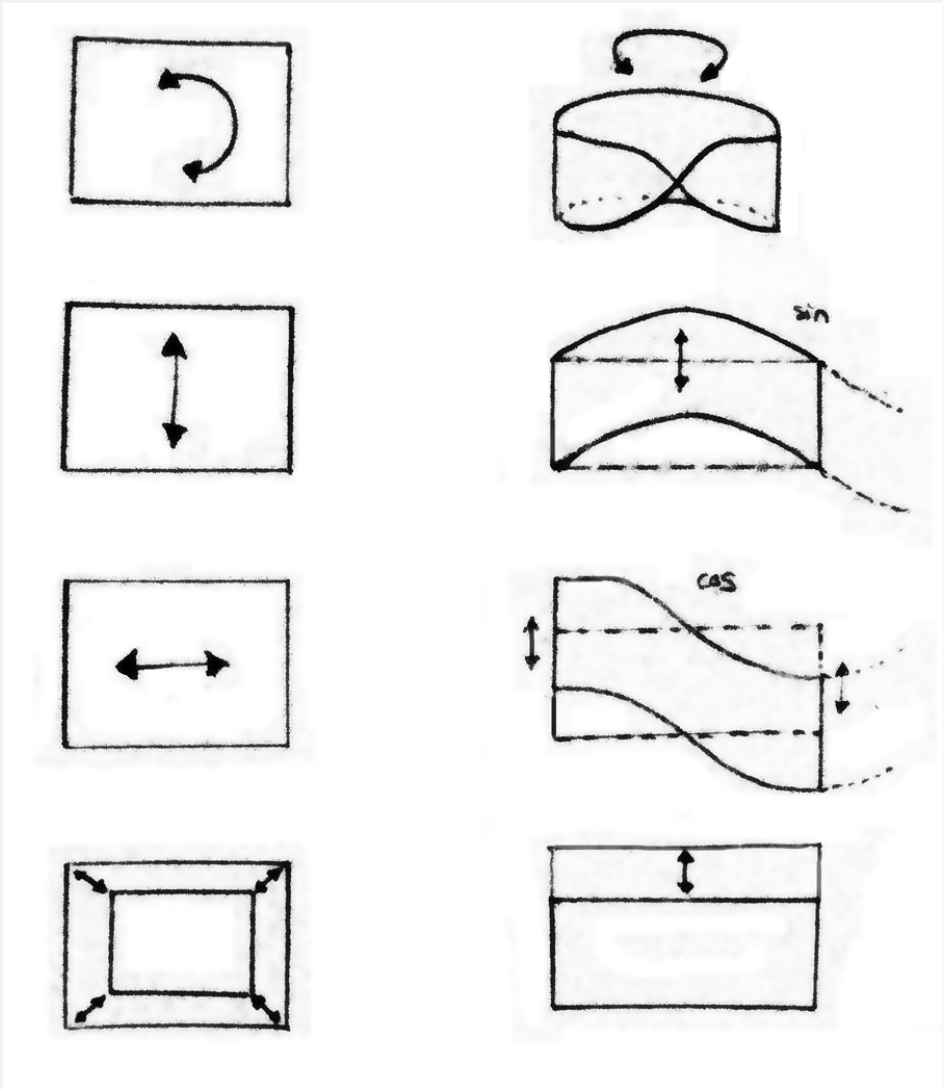
+



Example: Möbius-Radon transformation

- The shift over the Möbius ring corrects for rotation;
- For other corrections, things aren't so easy.

Example: Möbius-Radon transformation



Current situation on mobile phones

- Lots of image retrieval systems;
- There exists an open source computer vision library for Symbian (cpp). Drawback: support on a limited number of phones;
- We have chosen J2ME because of its widespread availability.

Java 2 Mobile Edition

- We've built an application which takes a picture and we're able to manipulate that image;
- We're now designing a small computer vision library which supports the most basic computer vision algorithms.

What remains to be done

- Experiment with features; choose final set;
- Decide upon learning algorithm;
- Implement J2ME feature extraction methods;
- Develop real-world test environment.