Food Product Logo Recognition using SIFT and RANSAC algorithms

Ross Peters
CSCI 512 – Computer Vision
Colorado School of Mines
Project Motivation

• New market for “everyday” videos: short videos taken on smartphones
  – Snapchat, Vine, Meerkat handling massive amounts of video data
  – How to extract user information from these videos?

• Logo Recognition using Feature Detection
  – What types of logos to begin with?
  – Application-driven demonstration
Goals and Approach

• Make a program that inventories the contents of a pantry from a video input
• Use SIFT (Scale-Invariant Feature Transform) to detect features in the frames
• Determine True Matches using RANSAC (RANdom SAmple Consensus)
• Programming Language: MATLAB
Early Test Cases
Early Test Cases
LogoDetect Flowchart (Single Frame)

1. **LogoDetect START**
2. **Input Frame**
3. **Gray-Level Single-Precision Input Logo**
   - **Resize Input Logo to size of Input Frame**
   - **Convert Input Frame to Gray-Level Single-Precision Image**
   - **Extract SIFT features and descriptor from Input Frame**
     - **Match a homography between the sets of image points**
     - **Match corresponding interest points using the two SIFT descriptors**
       - **RMS_err < 2?**
          - **TRUE**
             - **Logo is in Input Frame**
             - **Logo is not in Input Frame**
          - **FALSE**
             - **Logo is in Input Frame**
             - **Logo is not in Input Frame**
   - **Extract SIFT features and descriptor from Input Logo**
4. **LogoDetect END**
Feature Detection using SIFT

• Scale-Invariant Feature Transform: Image Descriptor designed by David Lowe (1999)
  – Creates scale-space from gray-level image, finds local extrema, uses approx. Laplacian of Gaussian to choose points of interest (scale selection)
  – Choose region around point of interest, measure gradient within region, create histogram of gradients
  – Create image descriptors from array of histograms
SIFT Features on Test Cases
SIFT Features on Test Cases
Matching Homography with RANSAC

• Random Sample Consensus: Outlier detection method designed by Fischler and Bolles (1981)
  – Finds inliers in outlier-infested data for model fitting purposes
  – Uses uncertainties of data samples containing outliers

• How to estimate outlier probability?
  – Using MSAC discussed in class
Flowchart for Fitting Homography

START

Determine probability that residual errors were inliers

Compare the transformed points to the detected features on the input frame

END

Estimate the probability of outlier point match sets

Update estimate of inlier point matches, update iter_estim

Transform the logo points using the resulting homography matrix

Determine uncertainty in homography using residual error

Determine number of inlier point match sets in the feature data

Update best homography fit indices, increment iter_count

Transform all logo points using homography

Estimate the number of trials necessary for a sample of all inlier point match sets

iter_estim

iter_count

iter_count < iter_estim?

Y

Use best fit points from inlier matches to produce homography

N

rms_err

hom_mtx

iter_count
Results with Still Images

Some Inconsistencies in RANSAC (~80% Detection Rate)

Grid:

- Number of matching features: 93
- Final iterations used: 655
- RMS Error: 0.8311

Grid 2:

- Number of matching features: 93
- Final iterations used: 445
- RMS Error: 193.5266
Results with Still Images

Iteration estimation not being modified

```
sample_count 196  
nIterations 197  
Final iterations used: 197  
RMS Error: 1.5173
```

```
sample_count 58774  
nIterations 58775  
Final iterations used: 58775  
RMS Error: 121.9441
```
Project Challenges

• Development so far
  – Consistency in detection with non-deterministic RANSAC algorithm
  – Preventing detection of false positives
  – Logo detection with image blur, bad lighting, etc.

• Upcoming Challenges
  – Moving to videos and frames
  – How to detect two identical logos in same image?
Future Work

• Working with lower quality videos
  – How to deal with blurring and other distortion?

• Using a different Image Descriptor
  – SURF and BRISK: More speed, less reliability

• Moving over to C++ with OpenCV Framework
  – Eventually Obj-C/Java for Mobile Development

• Extrapolate more information from logos (orientation, quantity, design)
Acknowledgements

Companies whose logos I used:
- Starbucks, Quaker Oats, Coca-Cola, Dole, Kellogg, Snapchat, Dallas Cowboys, MATLAB

Thanks to Dr. Hoff and CSM EECS Department for support
Questions?