Willow Garage, OpenCV, ROS, And Object Recognition
ICRA Semantic Perception Workshop
Gary Bradski
garybradski@gmail.com
Outline

- What's Willow Garage
  - Perception is Hard
  - Open Source Computer Vision Library (OpenCV)
  - Point Cloud Library (PCL)
- Current Research Results
- (if time) Speculations on Perception
What is Willow Garage?

• It's a privately funded robotics institute/incubator

• Mission to revolutionize civilian robotics
  • Not just companies, but the whole industry
    - Strong support of open source: ROS, OpenCV, PCL
  • Spin off companies and products

• Focus is on sensor based adaptive robots
What is ROS?

- Meta operating system for robotics
- Obtain, build, write, and run code across multiple computers, and multiple robots
Software Products

- ROS (Robot Operating System)
  - [http://www.ros.org](http://www.ros.org)

- OpenCV (Open Source Computer Vision Lib)

- PCL (Point Cloud Library)
  - [http://www.pointclouds.org/](http://www.pointclouds.org/)
Hardware Products

PR2s, Turtlebots, Spinouts
Working On

Software:
- Higher level object and scene recognition
- People, pose and tracking
- Perception Apps Store

Hardware:
- "ROS Arm": Capable but cheap
- Capable but cheap 2D+3D Sensing
Outline

- What's Willow Garage
- Perception is Hard
  - Open Source Computer Vision Library (OpenCV)
  - Point Cloud Library (PCL)
- Current Research Results
- (if time) Speculations on Perception
Vision is Hard

• What is it?
  – Turning sensor readings into perception.

• Why is it hard?
  – It’s just numbers.

We perceive this:

But the camera sees this:

Maybe try gradients to find edges?
Use Edges? … It’s not so simple

- Depth discontinuity
- Surface orientation discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)
- Illumination discontinuity (e.g., shadow)

Slide credit: Christopher Rasmussen
To Deal With the Confusion,
Your Brain has Rules...
That can be wrong

OpenCV’s purpose is to help turn “seeing” into perception.
We will also use active depth sensing to “cheat”.
Outline

- What's Willow Garage
- Perception is Hard
- Open Source Computer Vision Library (OpenCV)
  - Point Cloud Library (PCL)
  - Current Research Results
  - (if time) Speculations on Perception
**OpenCV: Open Source Computer Vision Library**

- Launched in 1999 while I was at Intel Corp.
  - **Purpose:** To advance computer vision by creating a comprehensive, mostly real time infrastructure available to all.
- Free and Open Source, BSD license
- 3.5M downloads
- 45K member user group
- Supported by Willow Garage, Nvidia, Google
- Learning OpenCV book by O'Reilly has been the best seller in Computer Vision and Machine Learning for 3 years now.
OpenCV Overview:

General Image Processing Functions

Machine Learning:
- Detection,
- Recognition

Segmentation

Image Pyramids

Geometric descriptors

Transforms

Camera calibration, Stereo, 3D

Features

Utilities and Data Structures

Tracking

Matrix Math

Fitting

OpenCV.willowgarage.com

2000 algorithms

Gary Bradski
Machine Learning Library (MLL)

CLASSIFICATION / REGRESSION
(new) Fast Approximate NN (FLANN)
(new) Extremely Random Trees
CART
Naïve Bayes
MLP (Back propagation)
Statistical Boosting, 4 flavors
Random Forests
SVM
Face Detector
(Histogram matching)
(Correlation)

CLUSTERING
K-Means
EM
(Mahalanobis distance)

TUNING/VALIDATION
Cross validation
Bootstrapping
Variable importance
Sampling methods

http://opencv.willowgarage.com
OpenCV Contracting Group

- Software development and contract consulting for OpenCV
# OpenCV - What’s new, What's coming

## New in OpenCV
- Full support for Android
- Ever growing GPU port
- Direct Kinect support
- Full C++ and STL compatible interface
- Full python interface
- Features2D – detectors/descriptors
- New, more accurate calibration patterns
- Fast Approximate Nearest Neighbor learning

## What's Coming
- Port to iOS iPhone/iPad
- A processing flow graph:
  - Write in python output in C++ or run as a ROS node
  - Higher level components
- Interoperability with PCL
- Perception App store
C++:

double calcGradients(const IplImage *src, int aperture_size = 7)
{
    CvSize sz = cvGetSize(src);
    IplImage* img16_x = cvCreateImage(sz, IPL_DEPTH_16S, 1);
    IplImage* img16_y = cvCreateImage(sz, IPL_DEPTH_16S, 1);

    cvSobel( src, img16_x, 1, 0, aperture_size);
    cvSobel( src, img16_y, 0, 1, aperture_size);

    IplImage* imgF_x = cvCreateImage(sz, IPL_DEPTH_32F, 1);
    IplImage* imgF_y = cvCreateImage(sz, IPL_DEPTH_32F, 1);

    cvScale(img16_x, imgF_x);
    cvScale(img16_y, imgF_y);

    IplImage* magnitude = cvCreateImage(sz, IPL_DEPTH_32F, 1);
    cvCartToPolar(imgF_x, imgF_y, magnitude);
    double res = cvSum(magnitude).val[0];

    cvReleaseImage( &magnitude );
    cvReleaseImage(&imgF_x);
    cvReleaseImage(&imgF_y);
    cvReleaseImage(&img16_x);
    cvReleaseImage(&img16_y);

    return res;
}

C++:

double contrast_measure(const Mat& img)
{
    Mat dx, dy;
    Sobel(img, dx, 1, 0, 3, CV_32F);
    Sobel(img, dy, 0, 1, 3, CV_32F);
    magnitude(dx, dy, dx);
    return sum(dx)[0];
}

Python API: Optical Flow Features

import cv

>>> img = cv.LoadImageM("building.jpg", cv.CV_LOAD_IMAGE_GRAYSCALE)
>>> eig_image = cv.CreateMat(img.rows, img.cols, cv.CV_32FC1)
>>> temp_image = cv.CreateMat(img.rows, img.cols, cv.CV_32FC1)
>>> for (x,y) in cv GoodFeaturesToTrack(img, eig_image, temp_image, 10, 0.04, 1.0, useHarris = True):
...    print "good feature at", x,y
Android Port

Example: Panorama using an a-Phone. These have the same data structure as used in Streetview.

See “The Vegan Robot” http://theveganrobot.com/
Segmentation

- Background subtraction,

- **pyramid, mean-shift, graph-cut**

- Watershed

```c
void watershed(const Mat& image, Mat& markers)
```
GrabCut

void grabCut(const Mat& image, Mat& mask, Rect rect, Mat& bgdModel, Mat& fgdModel, int iterCount, int mode)

Graph Cut based segmentation
Integral images

- Fast calculation of rectangular regions

\[
\begin{align*}
P_0 &= \{y, x\} = \{4, 4\} \\
P_1 &= \{y, x + w\} = \{4, 7\} \\
P_2 &= \{y + h, x\} = \{6, 4\} \\
P_3 &= \{y + h, x + w\} = \{6, 7\}
\end{align*}
\]

Upright rectangle

\[
\begin{align*}
P_0 &= \{y, x\} = \{1, 5\} \\
P_1 &= \{y + w, x + w\} = \{3, 7\} \\
P_2 &= \{y + h, x - h\} = \{4, 2\} \\
P_3 &= \{y + w + h, x + w - h\} = \{6, 4\}
\end{align*}
\]

rotated rectangle

void integral()
void cvPyrDown(IplImage* src, IplImage* dst, IplFilter filter = IPL_GAUSSIAN_5x5);

void cvPyrUp(IplImage* src, IplImage* dst, IplFilter filter = IPL_GAUSSIAN_5x5);
Read two input images:

Mat img1 = imread(argv[1], CV_LOAD_IMAGE_GRAYSCALE);

Detect keypoints in both images:

// detecting keypoints
FastFeatureDetector detector(15);
vector<KeyPoint> keypoints1;
detector.detect(img1, keypoints1);

Compute descriptors for each of the keypoints:

// computing descriptors
SurfDescriptorExtractor extractor;
Mat descriptors1;
extractor.compute(img1, keypoints1, descriptors1);

Now, find the closest matches between descriptors from the first image to the second:

// matching descriptors
BruteForceMatcher<L2<float> > matcher;
vector<DMatch> matches;
matcher.match(descriptors1, descriptors2, matches);
Tracking

2D
CamShift();
MeanShift();
KalmanFilter::
calcOpticalFlowPyrLK()
*Also see dense optical flow:
calcOpticalFlowFarneback()*

3D
Posit();
SolvePnP();

Start with a kernel $K(x - x_i) = c k \left( \frac{x - x_i}{h} \right)$ approximation of a probability distribution $P(x) = \sum_i K(x - x_i)$. Focus on the gradient $\nabla P(x) = \sum_i \nabla K(x - x_i)$.

Let $g(x) = -\nabla P(x)$, the derivative of the kernel and we get:

$$\nabla P(x) = \sum_i \sum_j \psi_{ij}(x) \frac{x - x_i}{h}$$

Window size
Mean shift vector

Gary Bradski (c) 2008
Homography & Camera Calibration

findHomography(...)

See samples/cpp/calibration.cpp

3D view of checkerboard

Un-distorted image

\[
\begin{bmatrix}
  x' \\
  y' \\
  z'
\end{bmatrix} =
\begin{bmatrix}
  f & 0 & 0 & 0 \\
  0 & f & 0 & 0 \\
  0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix}
\]

\[
x = f \frac{X}{Z} \\
y = f \frac{Y}{Z} \\
p = M_{int} P_C
\]
Stereo

- Once the left and right cameras are calibrated internally (intrinsics) and externally (extrinsics), we need to rectify the images.
ML Lib Example:
Boosting: Face Detection with Viola-Jones Rejection Cascade

In samples/cpp, see:
Multicascaedclassifier.cpp
Useful OpenCV Links

OpenCV Wiki:
http://opencv.willowgarage.com/wiki

OpenCV Code Repository:
svn co https://code.ros.org/svn/opencv/trunk/opencv

New Book on OpenCV:
http://oreilly.com/catalog/9780596516130/

Or, direct from Amazon:

Code examples from the book:
http://examples.oreilly.com/9780596516130/

Documentation
http://opencv.willowgarage.com/documentation/index.html
Outline

- What's Willow Garage
- Perception is Hard
- Open Source Computer Vision Library (OpenCV)
- Point Cloud Library (PCL)
- Current Research Results
- (if time) Speculations on Perception
3D Processing: PCL

- Point Cloud Library

- http://pointclouds.org/

Misc, stats:

- 35 releases already (0.1.x → 0.9.9)
- over 100 classes
- over 80k lines of code (PCL, ROS interface, Visualization)
- young library: only 12 months of development so far, but we had code lying around for 3-5 years
- external dependencies on eigen, cminpack, FLANN
Summary

PCL (Point Cloud Library) structure

PCL

- uses **SSE** optimizations for fast computations
- uses **OpenMP** and Intel **TBB** for parallelization
- data passing between modules using **shared pointers**
- ... GPU (...)
- is split into a collection of smaller, modular C++ libraries:
  - **libpcl_keypoints**: nD interest points
  - **libpcl_features**: nD feature descriptors
  - **libpcl_surface**: surface meshing/reconstruction techniques
  - **libpcl_filters**: point cloud data filters and smoothing
  - **libpcl_io**: I/O operations, 3D camera drivers (e.g., Kinect)
  - **libpcl_kdtree**: fast nearest neighbor operations
  - **libpcl_segmentation**: model segmentation operations
  - **libpcl_registration**: point cloud registration methods
- unit tests, examples, tutorials (!)
PCL Architecture

```
pcl::Feature<PointT> feat;
feat = pcl::Normal<PointT> (input);
feat = pcl::FPFH<PointT> (input);
feat = pcl::BoundaryPoint<PointT> (input);
...
feat.compute (&output);
...```
PCL: Processing Graphs

PPG: Perception Processing Graphs

- Clusters (1..N) (PointIndices)
- Indices (rest) (PointIndices)
- Projected inliers (PointCloud)
- Input (PointCloud)
- Normals (PointCloud)
- Inliers (PointIndices)
- Model (ModelCoefficients)
PCL: Filtering by depth

```cpp
p.setInputCloud (data);
p.FilterLimits (0.0, 0.5);
p.SetFilterFieldName ("z");
```
PCL: Finding Normals

```cpp
p.setInputCloud (data);
ps.setInputNormals (normals);
p.SetRadiusSearch (0.01);
```
PCL: Filtering by surface curvature

Point Cloud colored by depth:

Point Cloud colored by surface curvature:
PCL:
Using 3D features to classify surface types

3D Features are used to segment surface types:
Outline

- What's Willow Garage
- Perception is Hard
- Open Source Computer Vision Library (OpenCV)
- Point Cloud Library (PCL)

Current Research Results

(if time) Speculations on Perception
OpenCV - Recent TOD* (Textured Object Detection)

- 2D descriptors and detectors in 3D constellation using Kinect depth
- Bag of words to propose objects
- 3D to 3D fit to confirm recognition and
- Yield object pose in 6 degrees of freedom

* Similar to David Lowe’s work as well as MOPED (developed by Srinivasa Siddhartha, et. al.)
A TOD Result

See my Solutions in Perception Challenge:
http://opencv.willowgarage.com/wiki/SolutionsInPerceptionChallenge

Effort to establish what are solved problems in machine perception
And to drive solutions in new areas.
New Feature: ORB

- ORB (Oriented Brief) is a combination of a
  - Fast detector and
  - Brief descriptor

- FAST:
  - With reference to a central pixel “P” -- Interest points are detected as >= 12 contiguous pixel brighter than P in a ring of radius 3 around P.

Edward Rosten and Tom Drummond, “Fusing points and lines for high performance tracking”, ICCV 2005
New Feature: ORB

- ORB (Oriented Brief) is a combination of a
  - Fast detector and
  - Brief descriptor

- BRIEF:
  - Create an integral image for rapid summation of patches
  - In a 31x31 area round an interest point,
  - Randomly create 256 9x9 pairs patches, call them $A_i, B_i$
  - For each pair, if $\n$, then set the corresponding bit to 1, else 0
  - The resulting 256 bit vector is the descriptor for the patch

Calonder M., Lepetit V., Strecha C., Fua P.: BRIEF: Binary Robust Independent Elementary Features. ECCV 2010
Oriented FAST

- We orient the Fast detector by taking image moments at the corner:

- Moments:

\[ M_{ij} = \sum_x \sum_y x^i y^j I(x, y) \]

- Corner orientation:

\[ c_x = \frac{M_{10}}{M_{00}}, \quad c_y = \frac{M_{01}}{M_{00}} \]

\[ C_{ori} = \tan^{-1}\left(\frac{c_y}{c_x}\right) \]

Ethan Rublee and Vincent Rabaud and Kurt Konolige and Gary Bradski, ORB: an efficient alternative to SIFT or SURF, ICCV 2011 (probably)
Sterable Brief

- We add sterability to BRIEF:

- **Problem**: The patches become correlated:

- **Solution**: We exhaustively (greedily) search for decorrelated BRIEF patterns
New Feature: ORB

E. Rublee, V. Rabaud, K. Konolidge, G. Bradski, “ORB: an efficient alternative to SIFT and SURF”. ICCV 2011 (Submitted)

- Performance
  - Speed 100x faster than SIFT, 10x Faster than SURF
- Viewpoint invariance
- Noise tolerance
Binarized Grid (BIG)

- Used sets of binarized features from different modalities to recognize objects (Stephen Hinterstoisser's idea).
- Here, we use a binarized grid of dominant orientations “DOT” for object recognition proposal.
Robot Challenge: Solutions in Perception

• We (and Stanford) used ORB on the textured object dataset.

• Our (un-entered) entry is “TOD” (Textured Object Recognition)
  • Will be running in demo on the show floor all week

• Robot Challenge: Solutions in Perception.
  • Tues: Competition
  • Wed: Top competitors use recognition for grasping on PR2
Viewpoint Feature Histogram (VFH)

- New feature: Viewpoint Feature Histogram (VFH):

- Gives recognition and pose

BiGG Proposes the model
VPH Disposes

This is the model:

We get a fast, scalable and accurate classification and pose system
In Press: ICRA 2011
BIG + VFH
Outline

- What's Willow Garage
- Perception is Hard
- Open Source Computer Vision Library (OpenCV)
- Point Cloud Library (PCL)
- Current Research Results
- (if time) Speculations on Perception
Must deal with Lighting Changes ...

Which square is darker?

Checker-shadow illusion:
The squares marked A and B are the same shade of gray.
Use context to stabilize colors
Must deal with Lighting Changes ...

哪方较暗？

棋盘阴影错觉：标为A和B的方块是同一种灰色。
Lighting

Perception of surfaces depends on lighting assumptions
The Brain Assumes 3D Geometry

Perception is ambiguous … depending on your point of view!
Consequence of Projective Imaging: **Parallel lines meet**

- There exist vanishing points
Consequences* for **YOUR** Perception

Visual Metrics are Strange

Same size things get smaller, we hardly notice…

Parallel lines meet at a point…

Vergence Implies a Logarithmically Compressed Perceptual Space

Perception must be mapped to a space variant grid

Logarithmic in nature
Questions?