Open Source
Purpose and Models
Open Source Parts and Wholes
Gary Bradski
Senior Scientist, Willow Garage
Consulting Professor: Stanford CS Dept.

http://opencv.willowgarage.com
www.willowgarage.com
www.itseez.com
Outline

• Purpose of Open Source
• What is Willow
• What is ROS
• What is OpenCV
• OpenCV Quick Tour
• New in OpenCV
• Recent Work
• Solved Problems in Vision Contest
What is the Purpose of Open Source?

• Yell out a few ideas ...
The Arc of Computer Science

• Ever more encapsulation of ability in tools.
• “If I have seen further, its is because I have stood on the shoulders of giant code bases.”
What is the Purpose of Open Source?

My take:
1. The purpose is to accelerate human capability
2. To maintain that acceleration over time

• ... mainly through the use of easy recombination
Implication

**Purpose of Open Source:**
1. To accelerate human capability
2. To maintain that acceleration over time
   - ... mainly through the use of easy recombination

1. Open source need not be free
2. But it must be cheap
3. And impose the same “burden” on everyone
   - In a way that encourages positive feedback
Models?

<table>
<thead>
<tr>
<th>Purpose of Open Source:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. To accelerate human capability</td>
</tr>
<tr>
<td>2. To maintain that acceleration over time</td>
</tr>
<tr>
<td>• ... mainly through the use of easy recombination</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Implication:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Open source need not be free</td>
</tr>
<tr>
<td>2. But it must be cheap</td>
</tr>
<tr>
<td>3. And impose the same “burden” on everyone</td>
</tr>
<tr>
<td>• In a way that encourages positive feedback</td>
</tr>
</tbody>
</table>

• Anyone?
Models?

Purpose of Open Source:
1. To accelerate human capability
2. To maintain that acceleration over time
   • ... mainly through the use of easy recombination

Implication:
1. Open source need not be free
2. But it must be cheap
3. And impose the same “burden” on everyone
   • In a way that encourages positive feedback

My Take:
1. Compensation drives innovation and persistence
2. The “Song play model” on radio might work
   • Maybe a “fee” on all products of 1%
   • If a product uses open source code, the 1% goes to the open project in proportion to its use.
   • Companies that contribute back, become part of the that projects paid contributor pool.
Algorithms, not Code

Purpose of Open Source:
1. To accelerate human capability
2. To maintain that acceleration over time
   • ... mainly through the use of easy recombination

Implication:
1. Open source need not be free
2. But it must be cheap
3. And impose the same “burden” on everyone
   • In a way that encourages positive feedback

My Take:
1. Compensation drives innovation and persistence
2. The “Song play model” on radio might work
   • Maybe a “fee” on all products of 1%
   • If a product uses open source code, the 1% goes to the open project in proportion to its use.
   • Companies that contribute back, become part of the that projects paid contributor pool.

We want finer grain combinations
1. Algorithms, not whole packages
2. Plug ins, easy to use, fast to compile
Outline

• Purpose of Open Source

• What is Willow
• What is ROS
• What is OpenCV
• OpenCV Quick Tour
• New in OpenCV
• Recent Work
• Solved Problems in Vision Contest
Willow Garage ... and Open Source
What is Willow Garage?

• A Robotics Research Institute/Incubator
  – Mobile Human-Centric Robotics
• Turning Robotics into a Software Problem:
  – Creating Open, Free (BSD) Infrastructure
    • Software ROS, OpenCV
  – Building Robust Robotics Platforms
    • PR2, Texai
• Working with many different research groups
• Located on Willow Road in Menlo Park
• Now ~55 Full Time, ~15 Contract, ~20 interns
  – Plus postdoc and visiting faculty
Business Plan

• Same as everyone else’s:

We’re somewhere here

Gary Bradski.
http/OpenCV.willowgarage.com
Business Plan

1. Become the “Linux” of Robotics
   – Through funded open source development and
   – Unbeatable HW platform distributed free to select research groups who contribute open code back.

2. Foster an Industry, not just a Company
   – Wide external collaboration/infrastructure
   – We won’t “own” it all
   – Faster and faster development cycles (from months to a week now see youtube videos)

3. With Learned Expertise, Spin off Verticals
## Business Plan

1. Become the “Linux” of Robotics
   - Through funded open source development and
   - Unbeatable HW platform distributed free to select research groups who contribute open code back.

2. Foster an Industry, not just a Company
   - Wide external collaboration/infrastructure
   - We won’t “own” it all
   - Faster and faster development cycles (from months to a week now see youtube videos)

3. With Learned Expertise, Spin off Verticals

1. Ros is driving easy recombination

2. We practice this via short 1-2 week development sprints and “hackathons”.

Outline

- Purpose of Open Source
- What is Willow
- What is ROS
- What is OpenCV
- OpenCV Quick Tour
- New in OpenCV
- Recent Work
- Solved Problems in Vision Contest
ROSL
Robot Operating System
What is ROS?

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

I. computational graph
   - how programs run: master, node, topic, message, service, parameter

II. file system
    - how programs are organized and built: packages, stacks, msg, srv

III. repositories
    - how files are distributed online: ros-pkg, wg-ros-pkg, tum-ros-pkg, sail-ros-pkg, ...

Gary Bradski.
http/OpenCV.willowgarage.com
(I) Architecture Overview

- distributed processes called **nodes**
- synchronous **services**
- asynchronous **topics**
- data is passed via ROS **messages**

![Diagram of message and service request between Node1 and Node2]
(1) Nodes

- principal programming entities in ROS
- they connect to each other and pass data via *topics*
- the discovery of who publishes on what topic is done via a ROS master

```
(0. I am publishing on /chatter)  
1. who publishes on /chatter?  
2. node Talker  
3. connect to Talker  
4. start sending messages  
```
(I) ROS Messages

- defined in package-name/msg/.msg files, sent over topics
- basic data types: int\{8,16,32,64\}, float\{32,64\}, string, time, duration, array[]

Point.msg
float64 x
float64 y
float64 z

- used in ROS services, defined in package-name/srv/* .srv

Service = Request msg + Response msg
(I) master, roscore

- **Master**
  - Name service for ROS
    - Nodes register *topics* and *services* with Master
    - Nodes find each other using Master
  - XML-RPC based
- **roscore**
  - Master
  - *parameter* server
  - *rospub*: logs */rospub* topic for debugging
(II) Packages & Stacks

- **Packages** = directories with a certain structure, can contain anything: *nodes, messages, tools*
  - in their most basic form:
    - `package_name/`
    - `package_name/Makefile`
    - `package_name/CMakeLists.txt`
    - `package_name/manifest.xml`

- **ROS core** = small, but **ROS world** = many packages !!!
- **Stacks** = collection of packages
  - in their most basic form:
    - `stack_name/`
    - `stack_name/package_name_1`
    - `stack_name/package_name_N`
    - `stack_name/stack.xml`
(III) Repositories

- collection of packages and stacks, hosted online
- many repositories (>20): Stanford, CMU, TUM, etc
A Model for Open Source?

- ROS puts a thing wrapper around code blocks
- Allows easy combination through messaging.
- Maybe a model for finer grain Open Source combinations?
Outline

• Purpose of Open Source
• What is Willow
• What is ROS
• What is OpenCV
• OpenCV Quick Tour
• New in OpenCV
• Recent Work
• Solved Problems in Vision Contest
OpenCV Overview:

General Image Processing Functions

Segmentation

Transforms

Machine Learning:
• Detection,
• Recognition

Geometric descriptors

Image Pyramids

Features

Camera calibration, Stereo, 3D

Utilities and Data Structures

Tracking

Matrix Math

Fitting

> 500 algorithms

Gary Bradski.
http://OpenCV.willowgarage.com

opencv.willowgarage.com

Robot support

500 algorithms
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
Gary Bradski.
http://opencv.willowgarage.com
OpenCV History

- **Original goal:**
  - Accelerate the field by lowering the bar to computer vision
  - Find compelling uses for the increasing MIPS out in the market

- **Timeline:**

- **Staffing:**
  - Climbed in 1999 to average 7 first couple of years
  - Starting 2003 support declined between zero and one with exception of transferring the machine learning from manufacturing work I led (equivalent of 3 people).
  - Support to zero the couple of years before Willow.
  - 5 people over the last year

- ~2.5M downloads
OpenCV Tends Towards Real Time

Comparison with other libs: Performance

Description
Test station: Pentium M, 1.7GHz
Libraries: OpenCV 1.0pre, IPP 5.0, LTI 1.9.14, VXL 1.4.0
2D DFT: Forward Fourier Transform of 512x512 image
Resize: 512x512->384x384 bilinear interpolation, 8-bit 3-channel image
Optical flow: 520 points tracked with 41x41 window, 4 pyramid levels.
Neural Net: mushroom benchmark from FANN
License

• Based on BSD license
• Free for commercial or research use
  – In whole or in part
  – Does not force your code to be open
  – You need not contribute back
    • We hope you will contribute back, recent contribution, C++
      wrapper class used for Google Street Maps*

* Thanks to Daniel Filip
Where is OpenCV Used?

• Google Maps, Google street view, Google Earth, Books
• Academic and Industry Research
• Safety monitoring (Dam sites, mines, swimming pools)
• Security systems
• Image retrieval
• Video search
• Structure from motion in movies
• Machine vision factory production inspection systems
• Robotics

*Well over 2M downloads*
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: http://www.amazon.com/Learning-OpenCV-Computer-Vision-Library/dp/0596516134

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

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

User Group (39717 members): http://tech.groups.yahoo.com/group/OpenCV/join
Outline

• Purpose of Open Source
• What is Willow
• What is ROS

• What is OpenCV

• OpenCV Quick Tour
• New in OpenCV
• Recent Work
• Solved Problems in Vision Contest
Canny Edge Detector

Gary Bradski.
http/OpenCV.willowgarage.com
Distance Transform

• Distance field from edges of objects

Flood Filling

Gary Bradski. http/OpenCV.willowgarage.com
Hough Transform

Gary Bradski, Adrian Kahler 2008

Gary Bradski.
http/OpenCV.willowgarage.com
Space Variant vision: Log-Polar Transform

Screen shots by Gary Bradski, 2005

Gary Bradski.
http/OpenCV.willowgarage.com
void cvPyrDown(IplImage* src, IplImage* dst, IplFilter filter = IPL_GAUSSIAN_5x5);

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

Chart by Gary Bradski, 2005

Gary Bradski
http/OpenCV.willowgarage.com
Thresholds

Source Image:

Binary Threshold:

Adaptive Binary Threshold:

Screen shots by Gary Bradski, 2005
Histogram Equalization

Low Dynamic Range Image and its Histogram

Histogram Equalized Image and its Histogram

Screen shots by Gary Bradski, 2005
Gary Bradski.
http/OpenCV.willowgarage.com
Contours

Gary Bradski.
http/OpenCV.willowgarage.com
Morphological Operations Examples

- Morphology - applying Min-Max. Filters and its combinations

\[ \text{Image } I \]
\[ \text{Erosion } I \odot B \]
\[ \text{Dilatation } I \oplus B \]
\[ \text{Opening } I \odot B = (I \odot B) \oplus B \]

\[ \text{Closing } I \bullet B = (I \oplus B) \odot B \]
\[ \text{Grad}(I) = (I \oplus B) - (I \odot B) \]
\[ \text{TopHat}(I) = I - (I \odot B) \]
\[ \text{BlackHat}(I) = (I \oplus B) - I \]

Gary Bradski.
http://OpenCV.willowgarage.com
Image textures

- Inpainting:
- Removes damage to images, in this case, it removes the text.

Gary Bradski.
http/OpenCV.willowgarage.com
Segmentation

- Pyramid, mean-shift, graph-cut
- Here: Watershed

Screen shots by Gary Bradski, 2005
Recent Algorithms: GrabCut

- Graph Cut based segmentation

Images by Gary Bradski, © 2010
Motion Templates (work with James Davies)

- Object silhouette
- Motion history images
- Motion history gradients
- Motion segmentation algorithm

Charts by Gary Bradski, 2005

Gary Bradski.
http/OpenCV.willowgarage.com
Segmentation, Motion Tracking and Gesture Recognition

Gary Bradski.
http/OpenCV.willowgarage.com
New Optical Flow Algorithms

#include <opencv2/opencv.hpp>

int main(...){
    ... CvCapture* capture = <...> ?
        cvCaptureFromCAM(camera_id) :
        cvCaptureFromFile(path);
    if( !capture ) return -1;
    for(;;) {
        IplImage* frame = cvQueryFrame(capture);
        if(!frame) break;
        // ... copy and process image
        cvCalcOpticalFlowPyrLK( ... )
        cvShowImage( “LkDemo”, result );
        c = cvWaitKey(30); // run at ~20-30fps speed
        if(c >= 0) {
            // process key
        }
    }
    cvReleaseCapture(&capture

lkdemo.c, 190 lines
(needs camera to run)

\[ I(x + dx, y + dy, t + dt) = I(x, y, t); \]
\[ -\frac{\partial I}{\partial t} = \frac{\partial I}{\partial x} \cdot (dx / dt) + \frac{\partial I}{\partial y} \cdot (dy / dt); \]
\[ G \cdot \partial X = b, \]
\[ \partial X = (\partial x, \partial y), G = \sum \begin{bmatrix} I_x^2, I_x I_y \\ I_x I_y, I_y^2 \end{bmatrix}, b = \sum I_t \begin{bmatrix} I_x \\ I_y \end{bmatrix} \]
Tracking with CAMSHIFT

• Control game with head

Screen shots by Gary Bradski, 2005
Projections

Affine (2x2)

- Parallelograms

Perspective (3x3) or “Homography”

- Trapazoids (Includes all of Affine)

Screen shots by Gary Bradski, 2005
Stereo … Depth from Triangulation

- Involved topic, here we will just skim the basic geometry.
- Imagine two perfectly aligned image planes:

\[ d = x^l - x^r \]

Depth “Z” and disparity “d” are inversly related:

Gary Bradski.
http/OpenCV.willowgarage.com
Stereo

• In aligned stereo, depth is from similar triangles:

\[
\frac{T - (x^l - x^r)}{Z - f} = \frac{T}{Z} \Rightarrow Z = \frac{fT}{x^l - x^r}
\]

• Problem: Cameras are almost impossible to align
• Solution: Mathematically align them:
Stereo Rectification

- Algorithm steps are shown at right:
- Goal:
  - Each row of the image contains the same world points
  - "Epipolar constraint"

![Diagram showing rectification process]

Gary Bradski. http/OpenCV.willowgarage.com
Recognition

Hand tracking + Gradient Histogram:

Gary Bradski.
http/OpenCV.willowgarage.com
Patch Matching

by Gary Bradski

Gary Bradski (c) 2008
http/OpenCV.willowgarage.com
Boosting: Face Detection with Viola-Jones Rejection Cascade

by Gary Bradski

by Viola Jones

Not face

Not face

Not face

Face

by Gary Bradski

Gary Bradski (c) 2008

http://OpenCV.willowgarage.com
Outline

- Purpose of Open Source
- What is Willow
- What is ROS
- What is OpenCV
- OpenCV Quick Tour
- New in OpenCV
- Recent Work
- Solved Problems in Vision Contest

Gary Bradski.
http/OpenCV.willowgarage.com
New C++ API: Usage Example

**Focus Detector**

**C:**

```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++:**

```cpp
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];
}
```

Gary Bradski.
http/OpenCV.willowgarage.com
Software Engineering

• Works on:
  – Linux, Windows, Mac OS

• Languages:
  – C++, Python, C

• Online documentation:
  – Online reference manuals: C++, C and Python.

• We’ve been expanding Unit test code

• Will soon standardize on cxx or Google’s test system.

• TEST COVERAGE:
New Directory Structure

- Re-Organized in terms of processing pipelines

- Code site: https://code.ros.org/gf/project/opencv/
  - Core
  - Calibration, features, I/O, img processing
  - Machine Learning, Obj. Rec
  - Python

- ~2.5M downloads
New OpenCV Conceptual Structure

- Modules
  - Object Recog.
  - Features2d
  - Calib3d Stereo
  - VO SLAM
  - Stitching
  - imgproc
  - CORE

- User Contrib
- Other Languages
  - Python
  - Lua
  - ffmpeg
  - ML, FLANN
  - HighGUI
- SSE, TBB, GPU, MPU

Gary Bradski.
http/OpenCV.willowgarage.com
1. Allow plug and play
2. Form coherent sets
3. Test and improve against existing techniques

• Maybe this is how we should allow different combinations from the existing OpenSource code projects.
• And/or maybe combine things in a ROS like way.
Organize OpenCV by Processing Stacks.
Example: Features2d

Detectors available
• SIFT
• SURF
• FAST
• STAR
• MSER
• GFTT (Good Features To Track)

Descriptors available
• SIFT
• SURF
• One way
• Calonder (under construction)
• FERNS

Gary Bradski.
http/OpenCV.willowgarage.com
Detector interface

class CV_EXPORTS FeatureDetector
{
public:
    void detect( const Mat& image, vector<KeyPoint>& keypoints, const Mat& mask=Mat() ) const
    {
        detectImpl( image, mask, keypoints );
    }

    virtual void read(const FileNode& fn) {};
    virtual void write(FileStorage& fs) const {};

protected:
    virtual void detectImpl( const Mat& image, const Mat& mask, vector<KeyPoint>& keypoints ) const = 0;

    static void removeInvalidPoints( const Mat& mask, vector<KeyPoint>& keypoints );
};
Descriptor interfaces

• For descriptors that can be represented as vectors in multidimensional space: **DescriptorExtractor** and **DescriptorMatcher**

• More general interface (one way, decision-tree-based descriptors): **GenericDescriptorMatch**
Instantiate DescriptorExtractor

Ptr<FeatureDetector> detector = createDetector( "FAST" );
Ptr<DescriptorExtractor> descriptorExtractor =
    createDescriptorExtractor( "SURF" );
Ptr<DescriptorMatcher> descriptorMatcher =
    createDescriptorMatcher( "BruteForce" );
vector<KeyPoint> keypoints;
detector->detect( img1, keypoints);
Mat descriptors;
descriptorExtractor->compute( img, keypoints, descriptors );
Run Descriptor Matcher

descriptorExtractor->compute( img1, keypoints1, descriptors1 );
descriptorExtractor->compute( img2, keypoints2, descriptors2 );
vector<int> matches;
descriptorMatcher->add( descriptors2 );
descriptorMatcher->match( descriptors1, matches );
Compare and Contrast using Testbench output

Gary Bradski.

http://OpenCV.willowgarage.com
Features 2D Use: Homography

- If you have a known planar object on the ground plane,
  - you can use it to map any other ground pt in the image to its \((X,Y,Z)\) point on the ground

```plaintext
getPerspectiveTransform(objPts,imgPts,H);
invert(H,H_invt);
```

We used this in the DARPA Grand Challenge to map the image road segmentation to a bird’s eye view obstacle map:

Gary Bradski and Adrian Kaehler: Learning OpenCV
## Features2D Use: Visual Odometry

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>frame</td>
<td>1</td>
</tr>
<tr>
<td>key</td>
<td>0</td>
</tr>
<tr>
<td>keyframes</td>
<td>1</td>
</tr>
<tr>
<td>from start</td>
<td>0.001m</td>
</tr>
<tr>
<td>covered</td>
<td>0.000m</td>
</tr>
<tr>
<td>inliers</td>
<td>319</td>
</tr>
<tr>
<td>outliers</td>
<td>10</td>
</tr>
<tr>
<td>time per frame</td>
<td>34ms</td>
</tr>
</tbody>
</table>

Gary Bradski.  
[http/OpenCV.willowgarage.com](http/OpenCV.willowgarage.com)
New Object Rec. Pipelines Coming

**Object Rec**
**Input**
- Image list
- Depth list

**Output**
- Rec.
- Score
- Segment.
- Pose

**Pose**
**Input**
- Image list
- Depth list
- Segment.

**Output**
- Pose

**Pose Refine**
**Input**
- Image list
- Depth list
- Segment.

**Output**
- Pose

**Recog. Fusion**
**Input**
- Rec. list
- Score list
- Cross Val.

**Output**
- Rec.
- Segment.
- Confiden.

**Attention**
**Input**
- Image list
- Depth list

**Output**
- Mask

**Object Train**
**Input**
- Image list
- Depth list

**Output**
- Models

Gary Bradski.
http/OpenCV.willowgarage.com
Outline

• Purpose of Open Source
• What is Willow
• What is ROS
• What is OpenCV
• OpenCV Quick Tour
  • New in OpenCV
• Recent Work
• Solved Problems in Vision Contest
First: The Sensors PR2 has

Wide angle camera in the forearm

Gary Bradski.
http/OpenCV.willowgarage.com
Dense Stereo from Textured Light

- Textured (not structured!) light for dense depth maps

**Willow Garage: Milestone 2**

- Robot will navigate in a crowded building without problem for 26 miles – “a marathon”
- Robot will find 10 outlets behind 10 doors chosen on a web map.
  1. Open doors if necessary
  2. Plug in and out
  3. In less than 1 hour.

---

Dataset | Accuracy (higher is better)
---|---
2x2 orange | 100%
2x1 orange | 100%
2x1 white hall | 97%

Gary Bradski.
http/OpenCV.willowgarage.com
Milestone 2: Door Handle Detection

1. Lidar scan finds door plane and bright spot finds the handle

2. Image based door handle detection:
   - Door handle detected from image using a boosted cascaded classifier of haar features
   - Handle locations clustered across several frames for a more robust detection
   - Stereo information
     - Used to eliminate false positives based on position and size priors
     - Used to compute final handle position

3. If both agree, it’s a handle

http/OpenCV.willowgarage.com
Detecting Outlets from Far Away

- Far outlet detection
  - Outlet candidates obtained by segmenting the disparity image
  - Wall pose computed from the base laser scan
  - List of outlet candidates is filtered using position, size and orientation priors
  - Each image patch is rectified to obtain a frontal view of the outlet
  - Outlet identity confirmed using a template matching algorithm

Rivz action movie at: http://pub1.willowgarage.com/~kwc/rviz-13744-5.mp4

Gary Bradski. http/OpenCV.willowgarage.com
Visual Servoing to Plug In: 2D Features + Geometry

Yeah, yeah: We no longer need to rely on color.
Milestone 2:

Yeah, yeah: We no longer need to rely on tapping


Autonomous Door Opening and Plugging In with a Personal Robot, ICRA 2010
We organize gradients in a way that allows extremely rapid search. This allows us to directly scale recognition with compute cycles (Similar to Stefan Hinterstoiser’s DOT at this conference).

Gary Bradski, 2010, very similar to Stefan Hinterstoiser’s Dominant Orientation Template
Binary Gradient Grid (BiGG)

A variant but very very fast feature

1. Take image gradients > small threshold
2. Binarize in 8 directions
3. OR together over a small window
4. Collect in a grid over an object
5. Only index a subset, skip by window size.
6. Take many templates
   – Memorize segmentation and pose relative to variant pose

Gary Bradski, 2010
Use of BiGG for Manipulation

CS 324: Perception for Manipulation

Gary Bradski.
http/OpenCV.willowgarage.com
Low Level Processing: Separating Planes, Finding Normals

- Using dense depth from stereo with textured light:

- Compute normals, remove planes:
For every pairs of points \((p_i, p_j)\) and their associated normals \((n_i, n_j)\), compute:

\[
\begin{align*}
    f_0 & = \langle v, n_j \rangle \\
    f_1 & = \frac{\langle u, p_j - p_i \rangle}{\|p_j - p_i\|} \\
    f_2 & = \|p_j - p_i\| \\
    f_3 & = \text{atan}(\langle w, n_j \rangle, \langle u, n_j \rangle)
\end{align*}
\]

\[
\begin{align*}
    i_{\text{hist}} & = \sum_{x=0}^{x \leq 3} \left[ \frac{f_x \cdot d}{f_{x_{\text{max}}} - f_{x_{\text{min}}}} \right] \cdot d^x
\end{align*}
\]
Fast Point Feature Histograms (FPFH) (1/5)

Basic Concepts

Re-formulate:

$$\text{FPFH}(p) = \text{SPF}(p) + \frac{1}{k} \sum_{i=1}^{k} \frac{1}{\omega_k} \cdot \text{SPF}(p_k)$$
Fast Point Feature Histograms (FPFH) (2/5)

Theoretical formulation
Point Feature Histograms (PFH) (1-4/5)

Points lying on different geometric primitives

Gary Bradski.
http/OpenCV.willowgarage.com
Surface Classification (8/8)

Classification results using FPFH and CRF: 97.36%
FPFH Classification

Classification results using FPFH and CRF: 98.27%

* Joint work with Gary Bradski

Gary Bradski.
http://OpenCV.willowgarage.com
Moving Towards Object Class Recognition
Global FPFHs (GFPFH)

Detecting and Segmenting Objects for Mobile Manipulation, Rusu, Radu Bogdan, Holzbach Andreas, Beetz Michael, and Bradski Gary, ICCV’09, S3DV Workshop
Object Class Recognition Architecture

- Flow Chart:

1. 3D Point Cloud
2. Normal estimation
3. Segmentation
4. Object cluster 1, ..., Object cluster n
5. FPFH estimation
6. Local Point Annotation
7. GFPFH estimation
8. Global Object Annotation
9. CRF model (layer-1)
10. SVM model (layer-2)

Gary Bradski.
http://opencv.willowgarage.com
GFPFH Classification

Classification results using GFPFH and SVM: 95.13%
Recognition

- Surface class results: 98.27%
- Object class results: 96.69%
FPFH Use in Manipulation
Latest Work: ViewPoint Feature Histogram (VPFH)

- For recognition and 6DOF pose
- ... 98% recognition of object and its pose (IROS 2010 In Press)

Gary Bradski. http/OpenCV.willowgarage.com
Latest Work

3D Pop Out from 2D

- ECCV 2010 In Press

(a) Test Image  (b) Detection  (c) Sparse 3D

(d) Model Fitting  (e) 3D Bounding Box  (f) Pop-Up
Outline

• Purpose of Open Source
• What is Willow
• What is ROS
• What is OpenCV
• OpenCV Quick Tour
• New in OpenCV

• Recent Work

• Solved Problems in Vision Contest
Working on: Object Database

- > 200 Objects
- Textured, transparent, opaque, translucent
- 3D Models
- Stereo depth maps
- Annotated with grasping points

Gary Bradski
http://opencv.willowgarage.com
Working on: Computer Vision challenges

- There are many data set challenges:
  - PASCAL VOC,
  - CalTech101
  - CalTech256 ...
  - CVPR semantic robot vision challenge

All of these explore the “False Positive” region of the ROC curve
I want to publically establish solved problems in vision

This corresponds to exploring the True Positive part of the curve

And push it to: “solved” for increasing classes of data:

Start with “easy” problems, advance from there
Working on: Solved Problems in Vision Contest

• Give people 3D Models
• Lots of imagery
• Stereo scans
• Limited object types (Lambertian, rigid)
• Can the community solve the recognition problem at 100%?
  – If yes, declare victory, post code and move on
  – If no, try again next time

Gary Bradski.
http/OpenCV.willowgarage.com
Questions?