SD-VBS:
The San Diego Vision Benchmark Suite

Sravanthi Kota Venkata, Ikkjin Ahn, Donghwan Jeon, Anshuman Gupta, Chris Louie, Saturnino Garcia, Serge Belongie, Michael B. Taylor

Computer Science and Engineering
University of California, San Diego

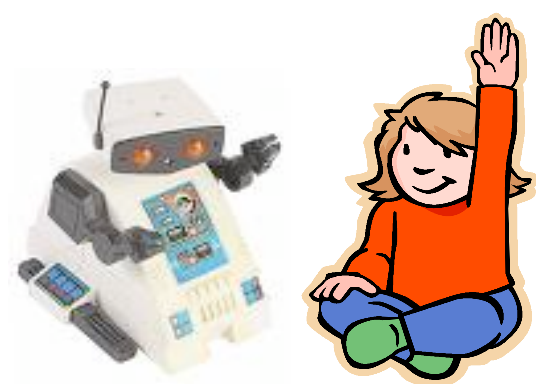
http://parallel.ucsd.edu/vision

Vision is an exciting application domain for many-core and multi-core systems

- "Enabling computers to see" will have a tangible and immediate impact on people's lives
- Limitless thirst for computation
  - Larger image sizes
  - More accurate analyses
  - Match and search against larger databases
  - Run the algorithm in real-time, or even super-real-time
- Full of parallelism to put those idle cores to work
  - i.e. drive user demand for future multi-core processors and keep Moore's Law going
- Enormous progress in computer vision research over the last decade, and more to come
  - Some problems are now even considered "solved."

A Few Examples of Vision's Current and Potential Impact on Our Lives

- Auto-focus cameras and cell-phones through face detection
- Help doctors perform surgery on patients thousands of miles away
- Allow planes to fly themselves (e.g., UAVs)
- Enable "cars that can't crash"
- Enable machines that automatically educate our kids (!)



SD-VBS: The San Diego Vision Benchmark Suite

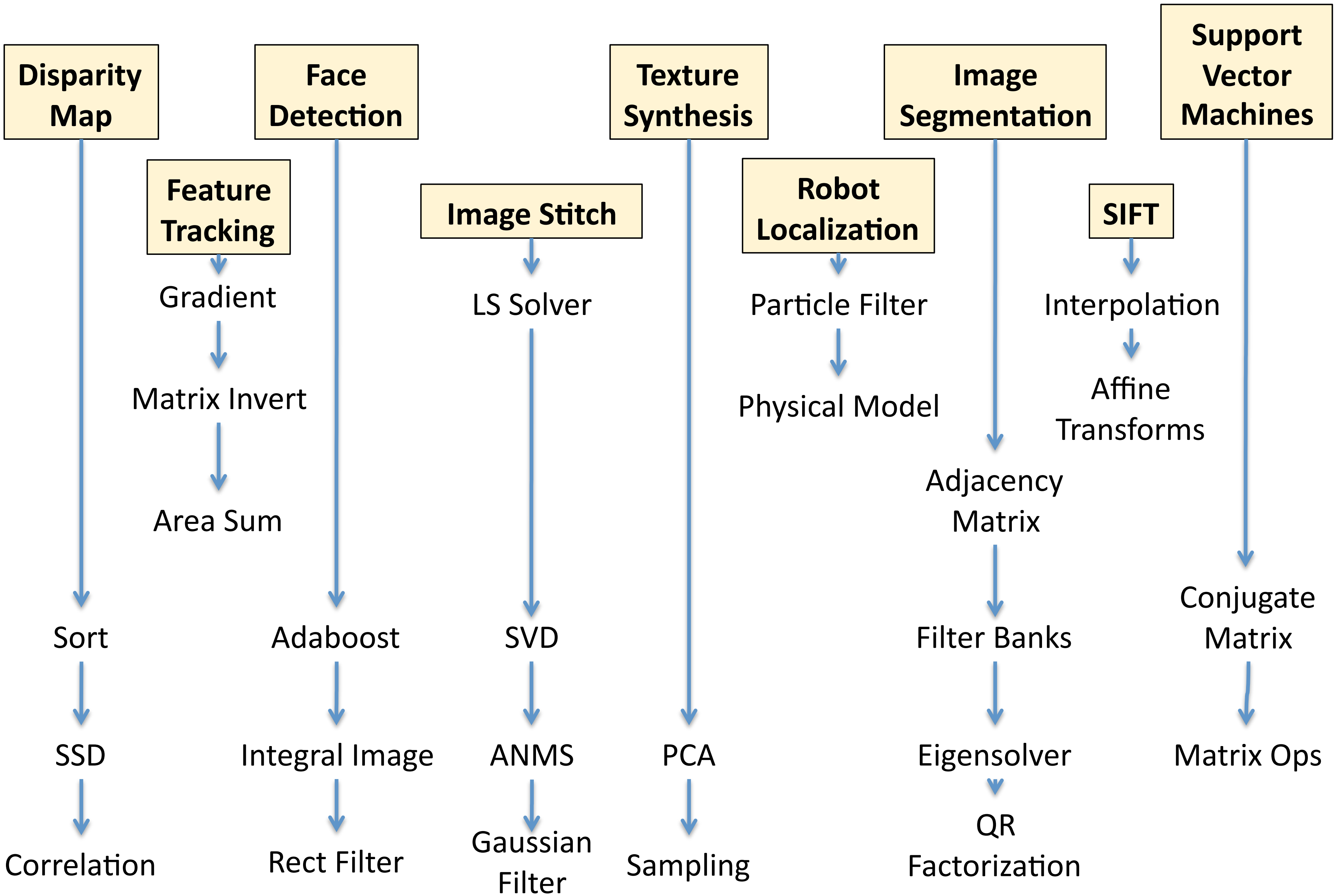
- Aim: Make vision more accessible to multi-core researchers through a comprehensive and easy to use suite.
- *9 benchmarks in 4 different areas.*

Benchmark	Concentration Area
Disparity Map	Motion, Tracking and Stereo Vision
Feature Tracking	Motion, Tracking and Stereo Vision
Image Segmentation	Image Analysis
Scale Invariant Feature Transform (SIFT)	Image Analysis
Robot Localization	Image Understanding
Support Vector Machines (SVM)	Image Understanding
Face Detection	Image Understanding
Image Stitch	Image Processing and Formation
Texture Synthesis	Image Processing and Formation

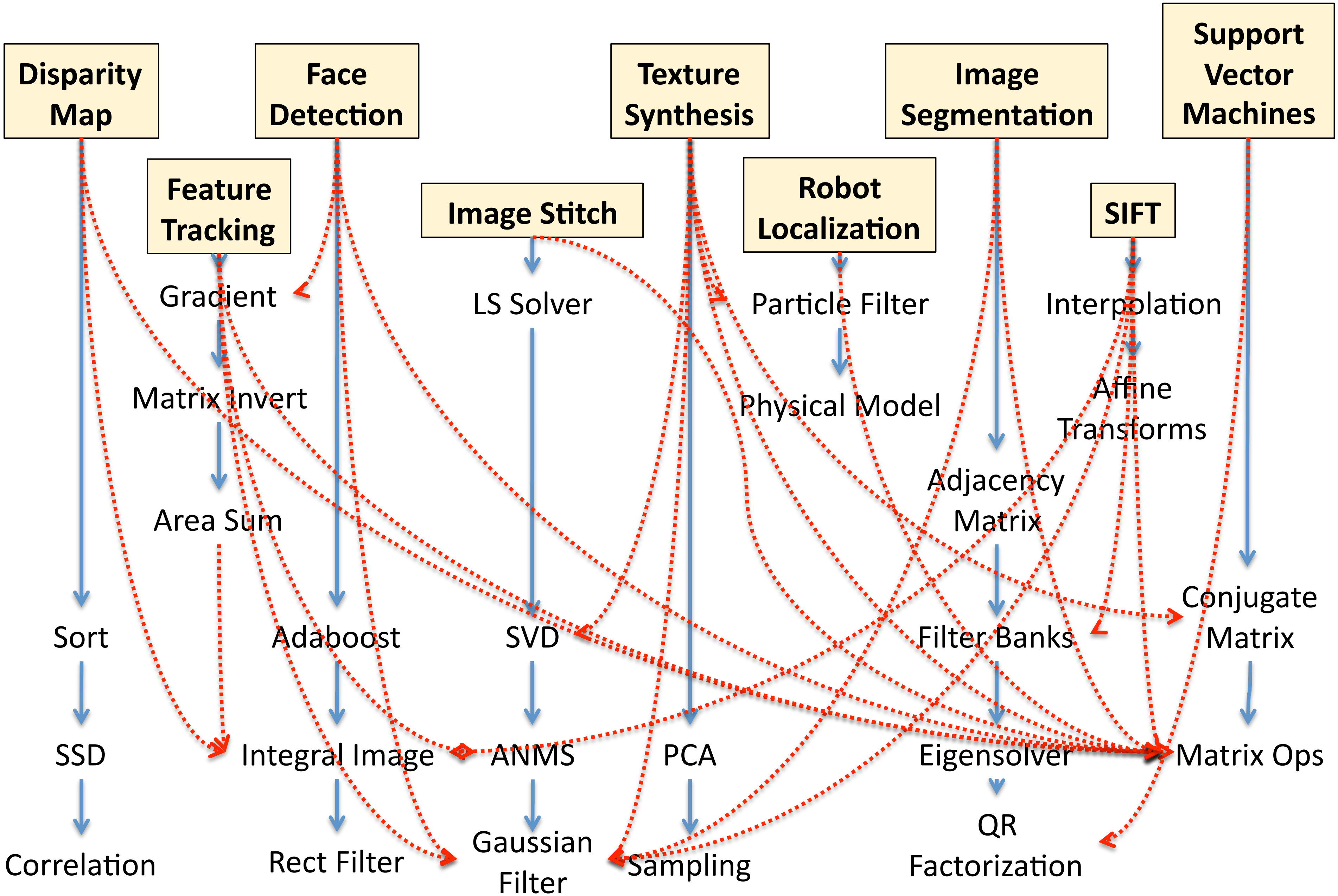
Design goals of SD-VBS

- Clean implementations for ease of analysis and transformation by researchers or their tools
  - "Clean" C and MATLAB versions
  - No dependence on custom libraries
- Comprehensive collection of representative benchmarks
- Low complexity of porting and parallelizing to different platforms

SD-VBS: 9 applications, > 25 kernels



Rich Reuse of Kernels in Vision



Overview of Talk

- Introduction to SD-VBS
- Vision Benchmarks Overview
  - foreach { Feature Tracking, Disparity, Image Stitch, SIFT,
    Segmentation, SVM, Robot Localization,
    Texture Synthesis } :

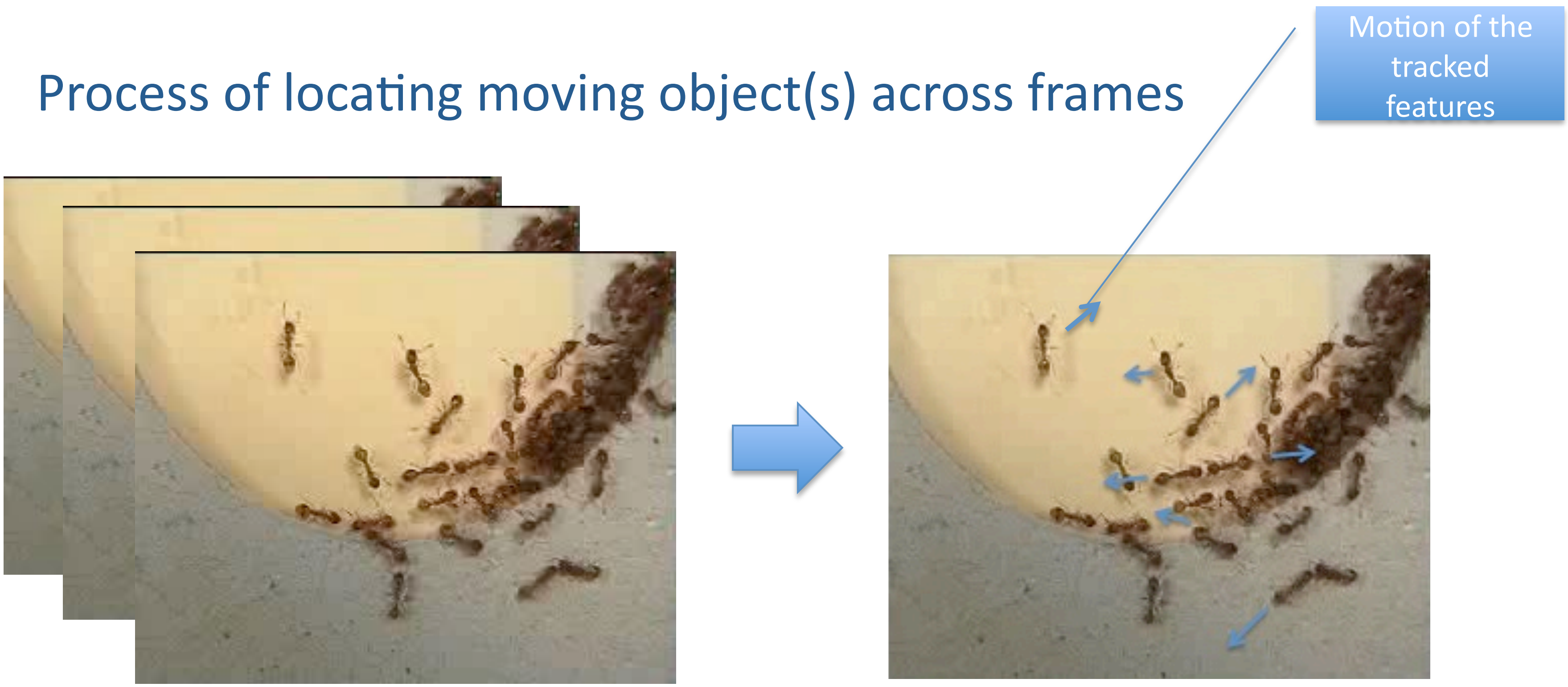
    Brief Demo

    Algorithm Description

    Analysis of Characteristics and Hotspots
- Results
- Related Work
- Conclusion

Feature Tracking: Overview

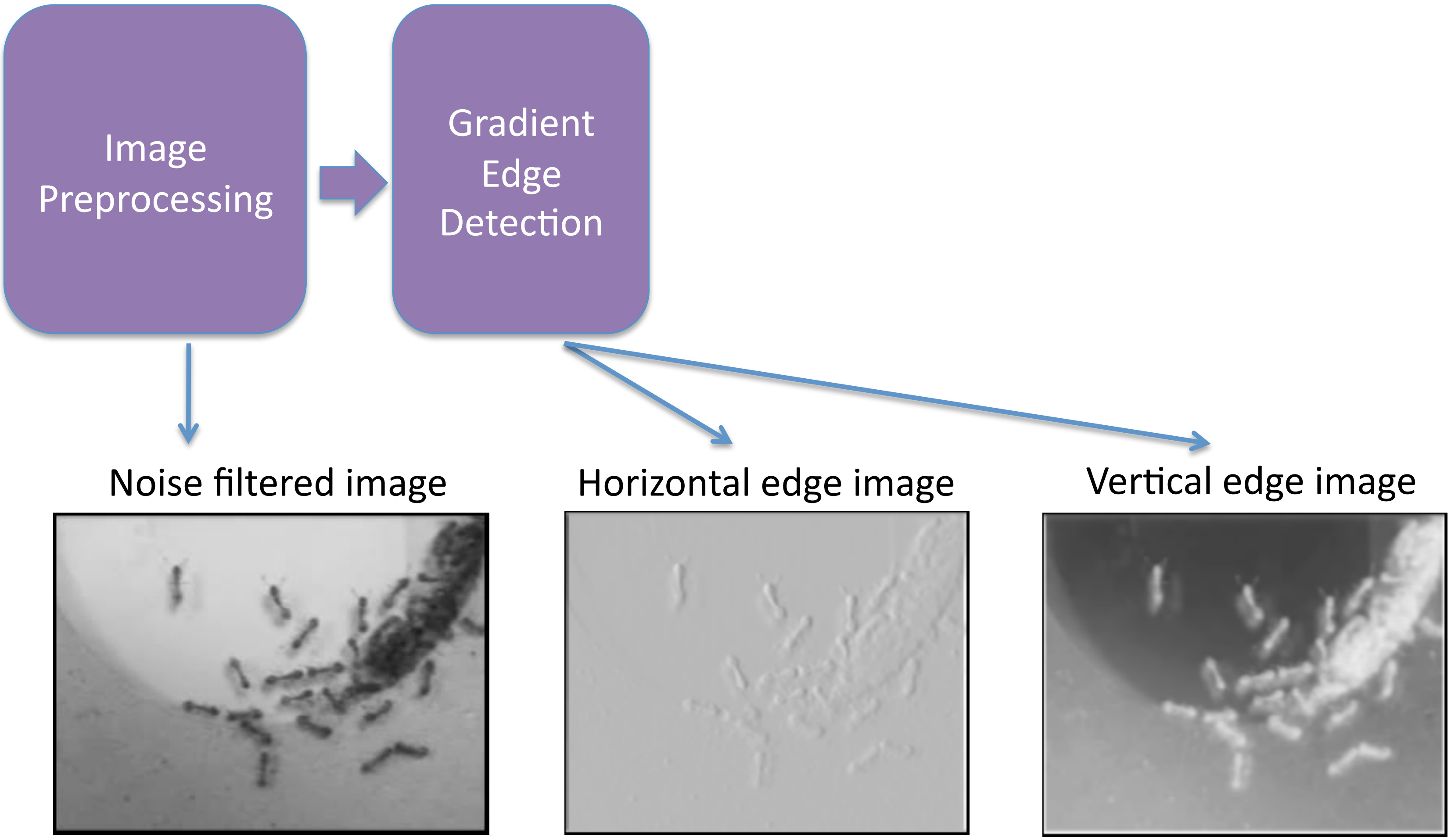
- Process of locating moving object(s) across frames



Applications:

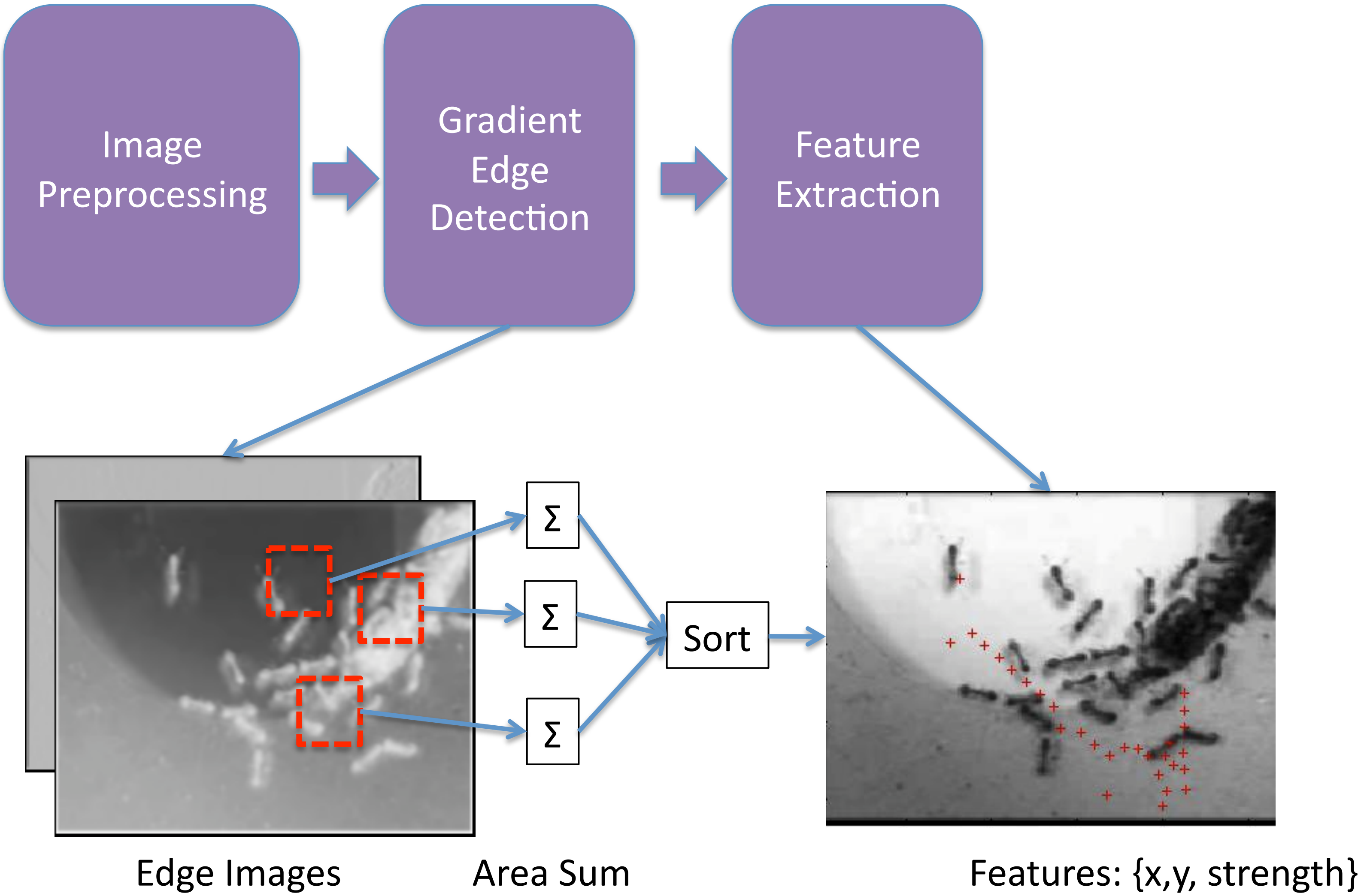
- Cruise Control
- Pedestrian Tracking
- Interactive Robots

Feature Tracking: Algorithm



Highly parallel: Each output pixel can be computed in parallel with others

Feature Tracking: Algorithm



Feature Tracking: Algorithm

Image Preprocessing → Gradient Edge Detection → Feature Extraction → Feature Matching

Frame 1 Features

< x, y, strgth>

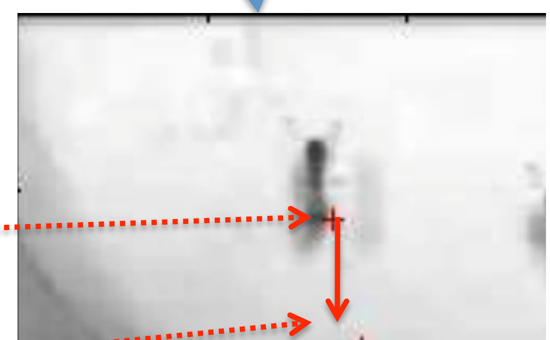
< 0,0, 25>
< … >

Frame 2 Features

< x, y, strgth>

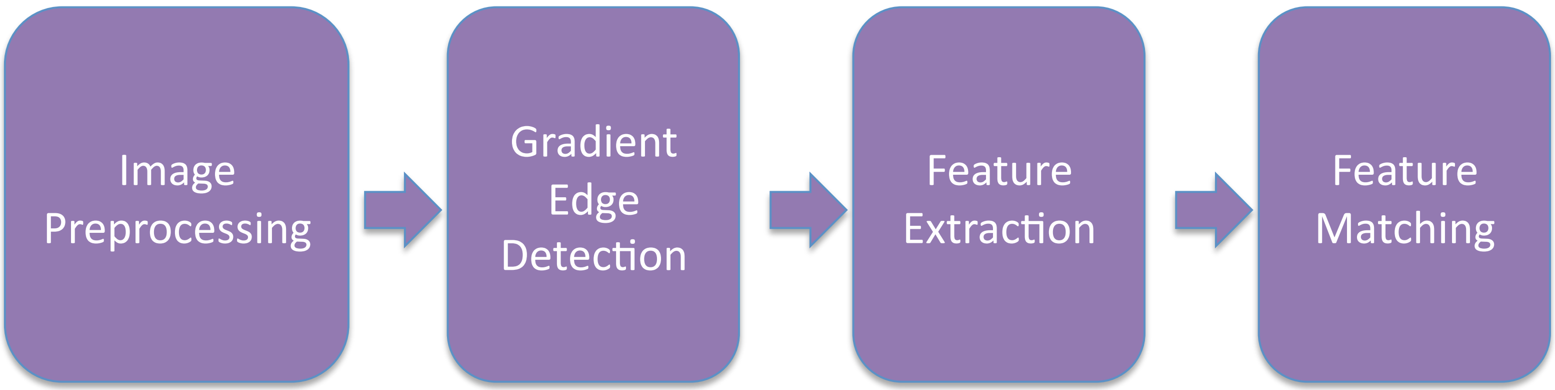
< 6,6, 21>
< 1,1, 12>
<0,2,22>

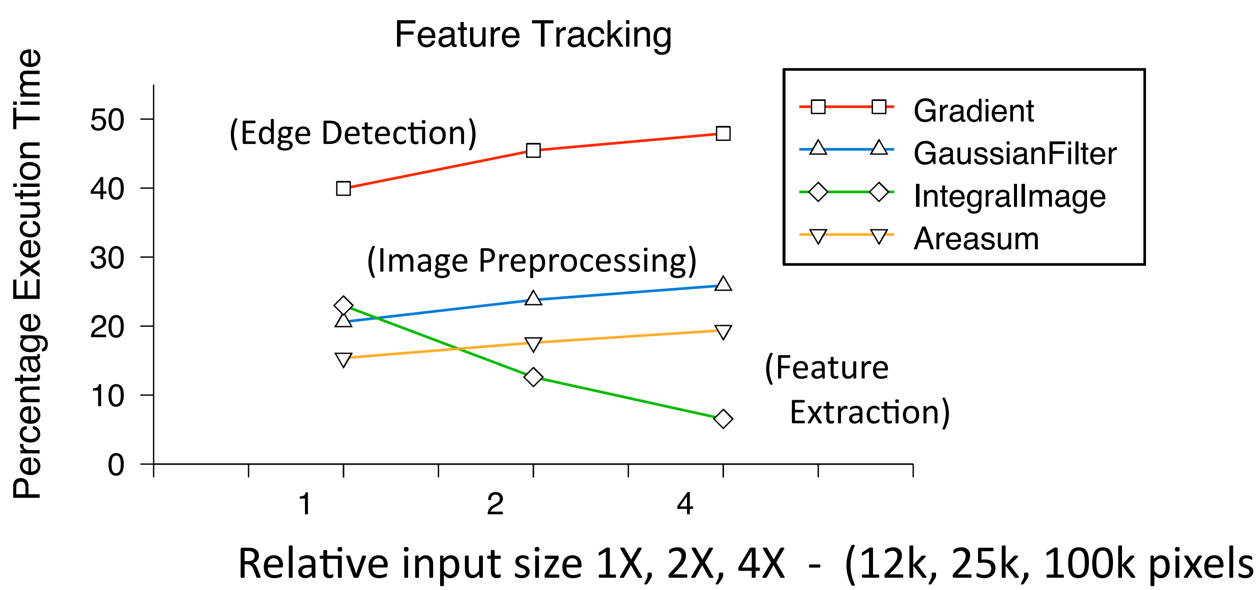
Select Nearby Features (e.g., within a range of 5x5) then pick one with closest strength.



Correlated, with Vector

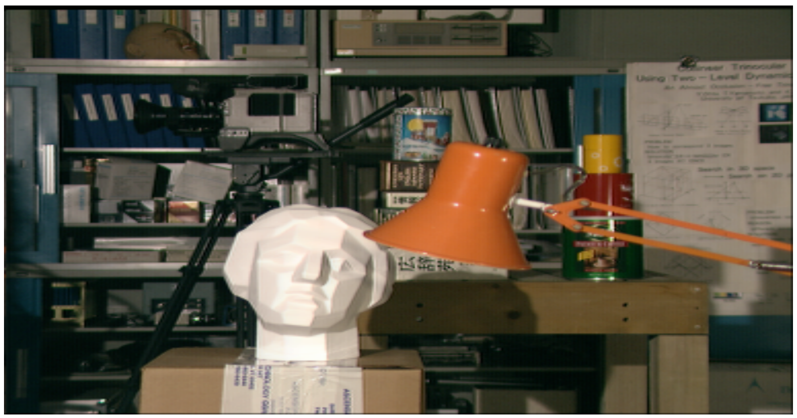
Feature Tracking: Algorithm



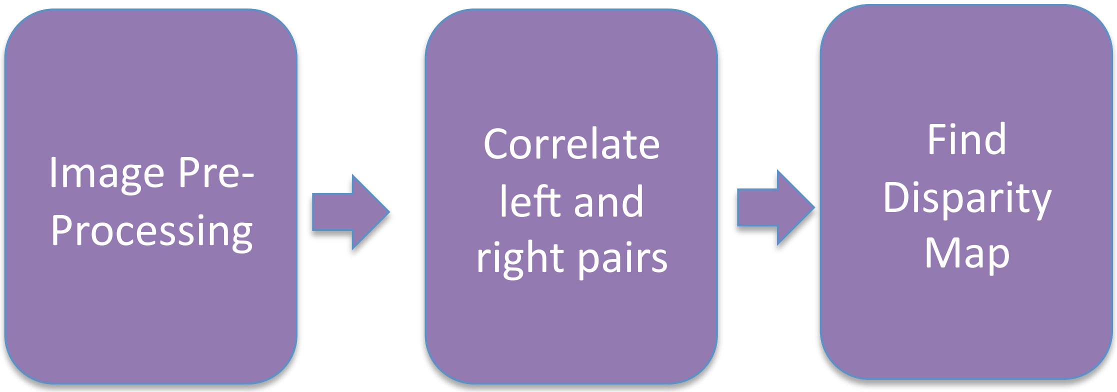


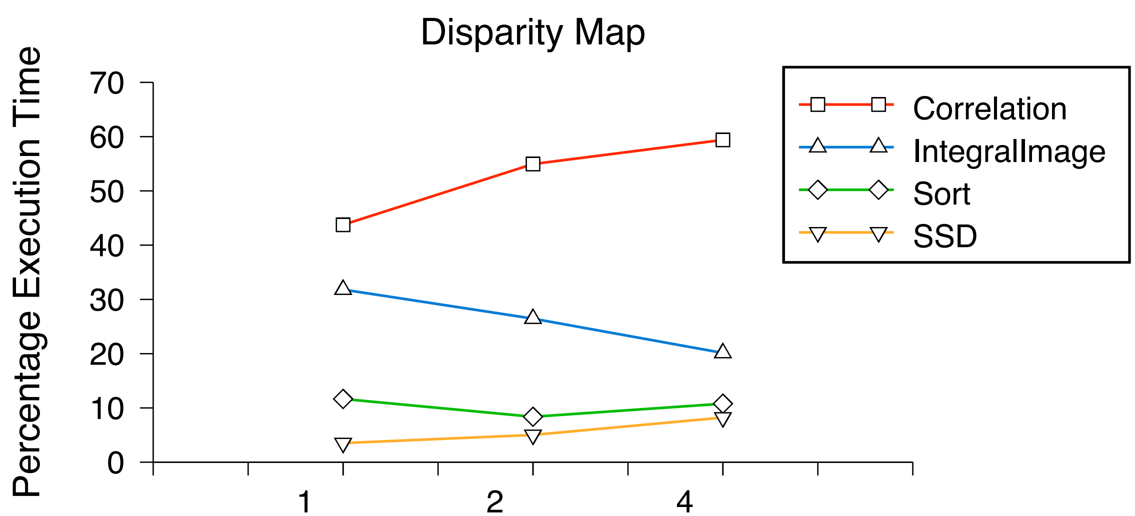
Disparity Map: Overview

- Computes relative positions of objects in a scene captured by stereo cameras – depth



Disparity Map: Algorithm





Relative input size 1X, 2X, 4X

Image Stitch: Overview

- Combining multiple images with overlapping view





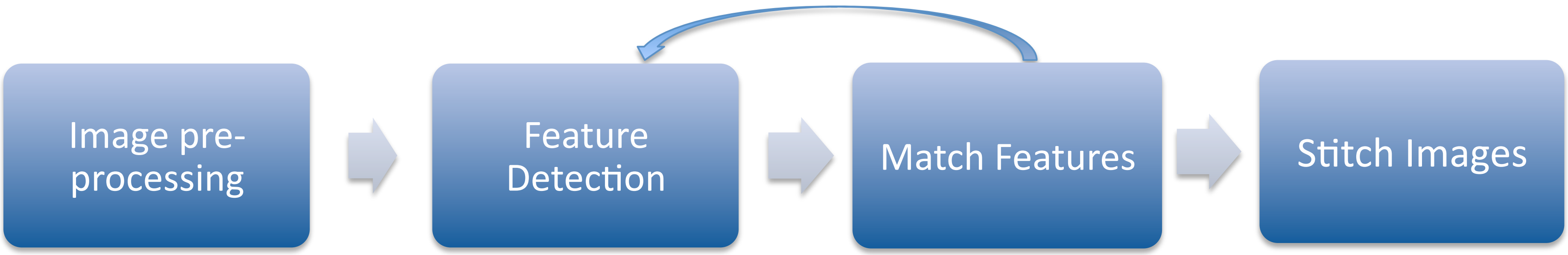
Image Stitch: Overview

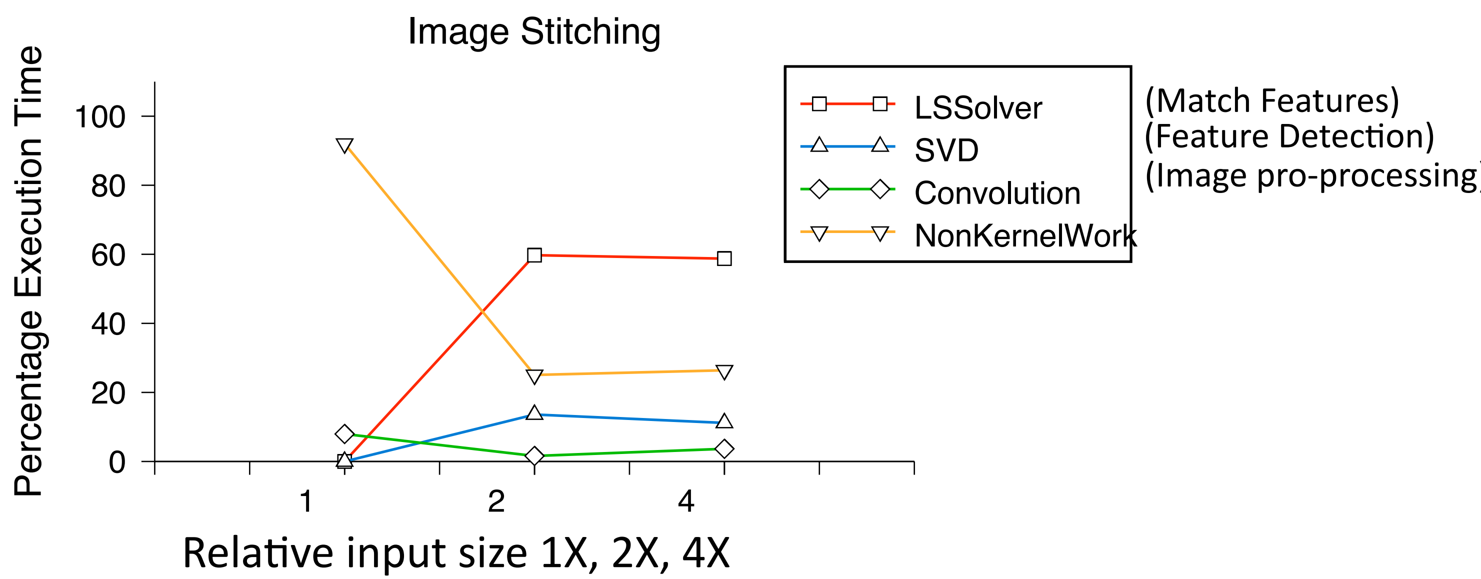
- Combining multiple images with overlapping view

Applications

- Movie making
- Google earth, GPS applications etc

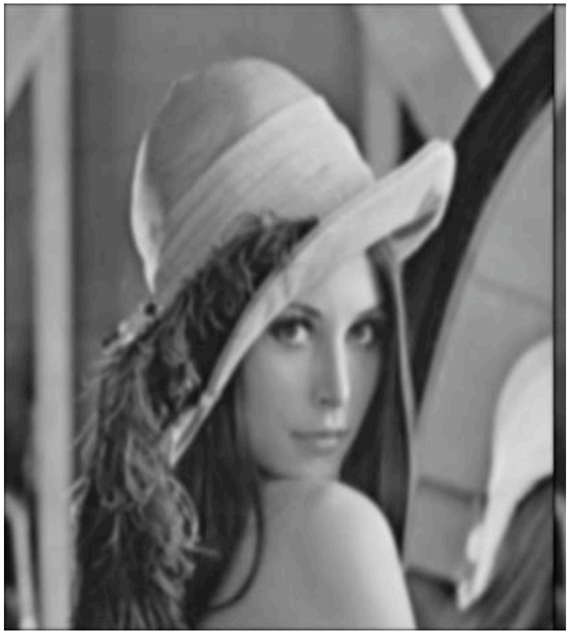
Image Stitch: Analysis





Scale Invariant Feature Transform (SIFT): Overview

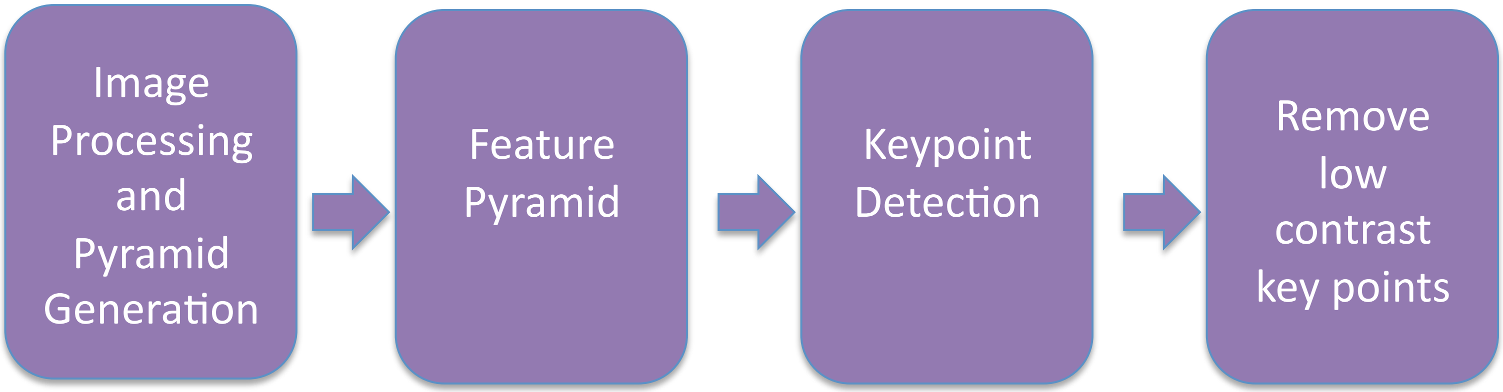
Detection and description of robust local image features

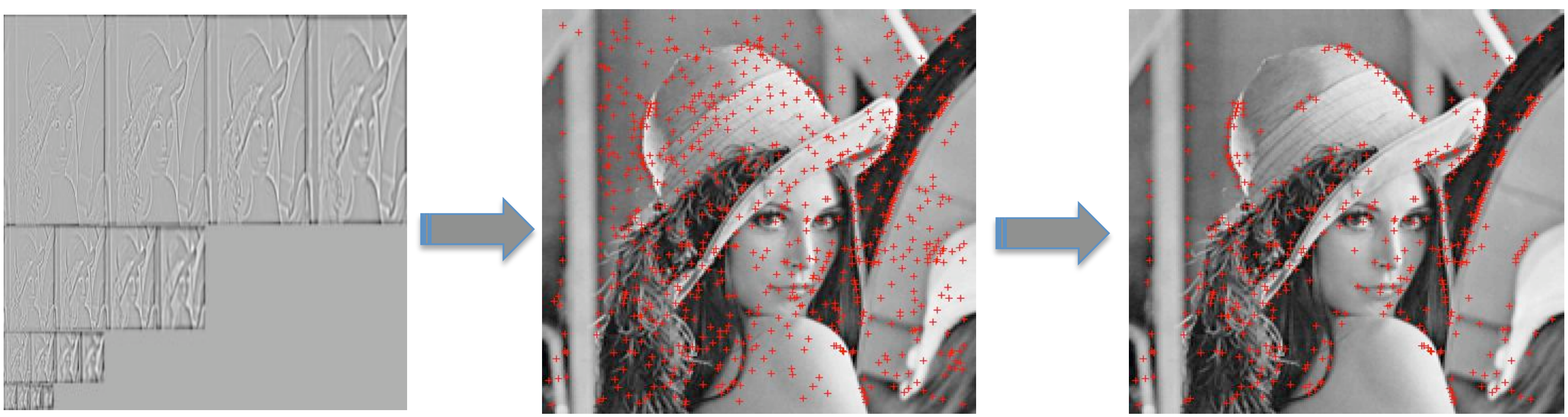


Applications

- Object recognition
- Motion Analysis
- Gesture Recognition

SIFT: Algorithm



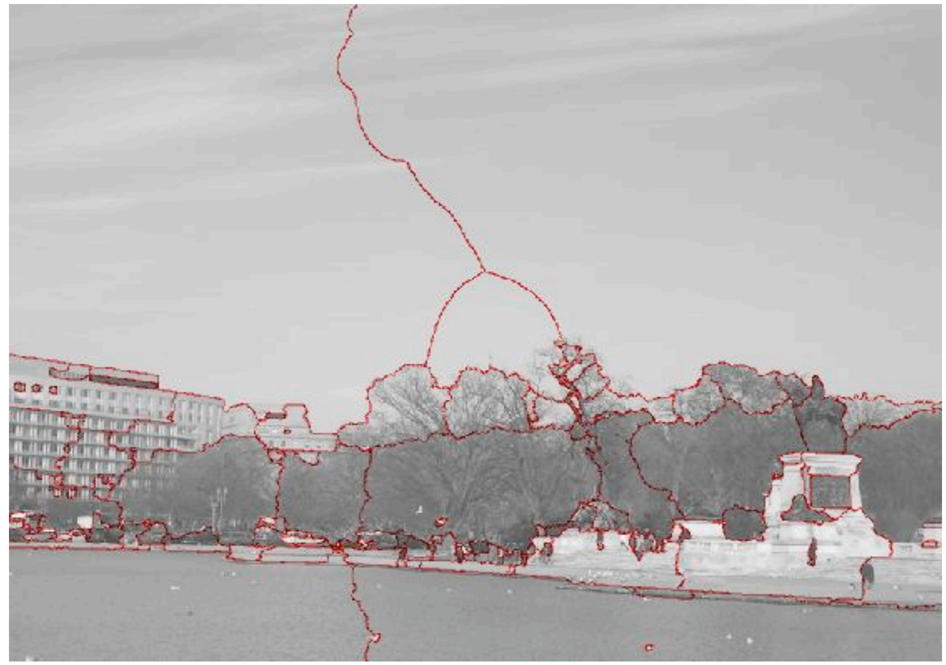


Difference of Gaussians Pyramid – Edge Images

Corner detection

Remove low contrast key points

Image Segmentation: Overview

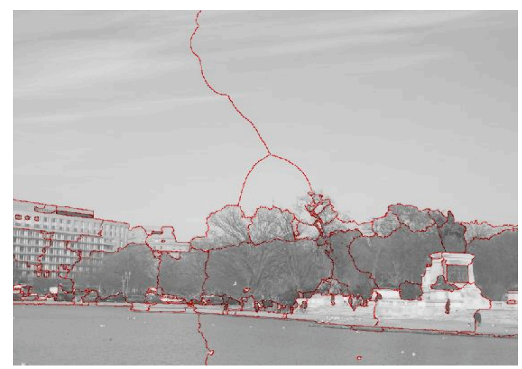


Process of partitioning an image into meaningful segments, typically used to locate objects and boundaries

Image Segmentation: Algorithm

Pyramid Generation and Gradient Image → Extract continuous edges → Extract high contrast edges → Mark discrete segments

Continuous edge image

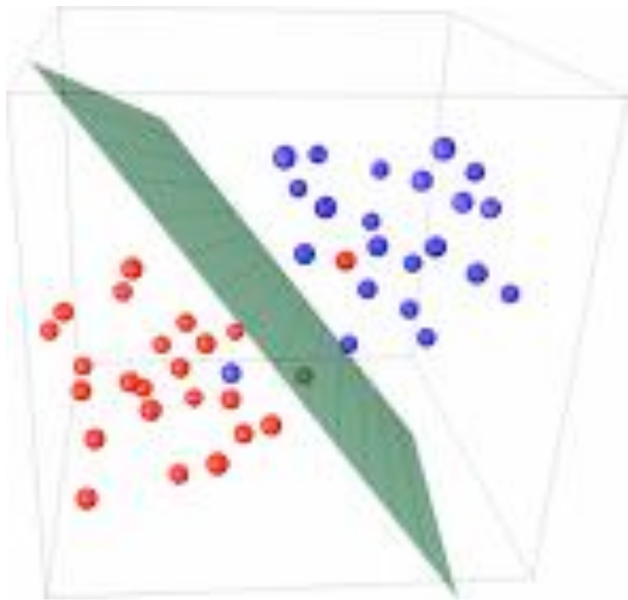


Segmented image

Using the continuous edge image and high contrast matrix, we mark segments

Support Vector Machine (SVM): Overview

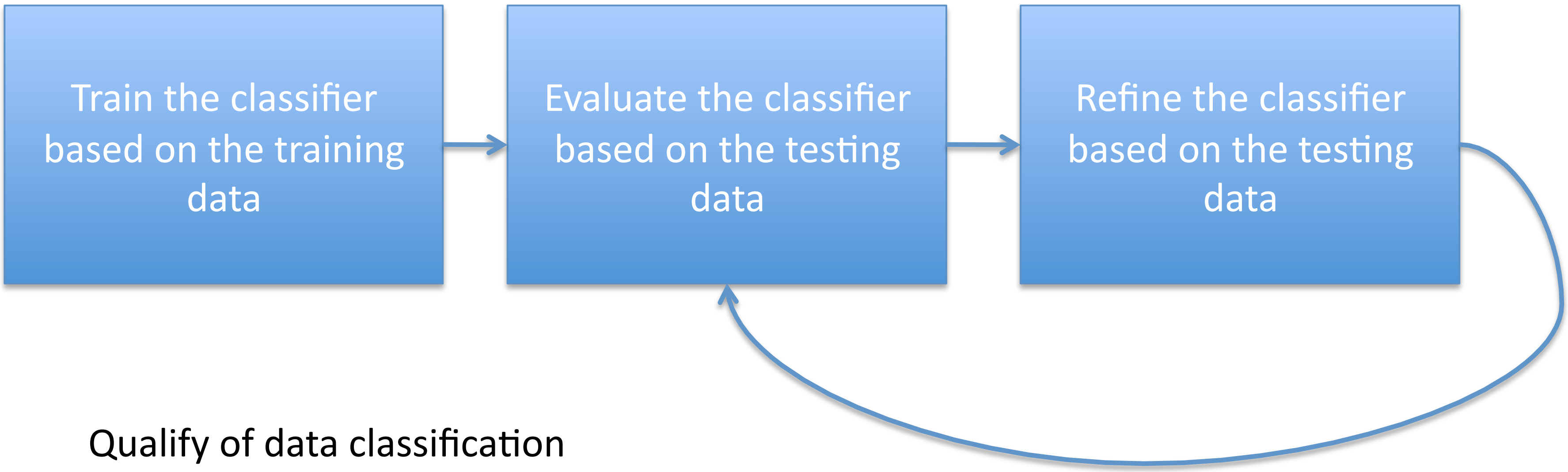
Separation of a given set of data into two categories with maximal geometric margin.



Applications

- Machine Learning
- Neural networks
- Data classification

SVM: Algorithm

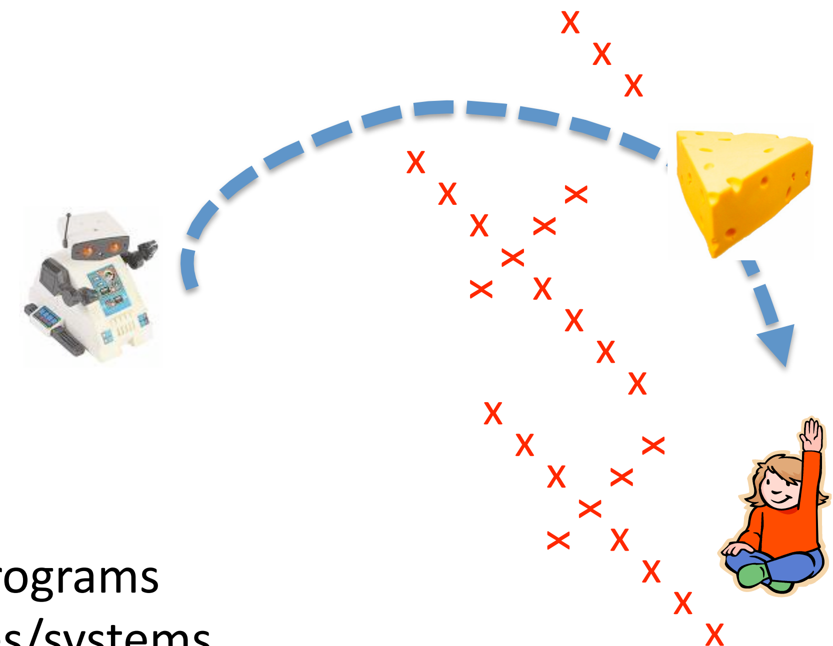


Qualify of data classification
Improves with number of
Iterations used in learning stage.

The execution time depends on the number of data points and number of iterations.

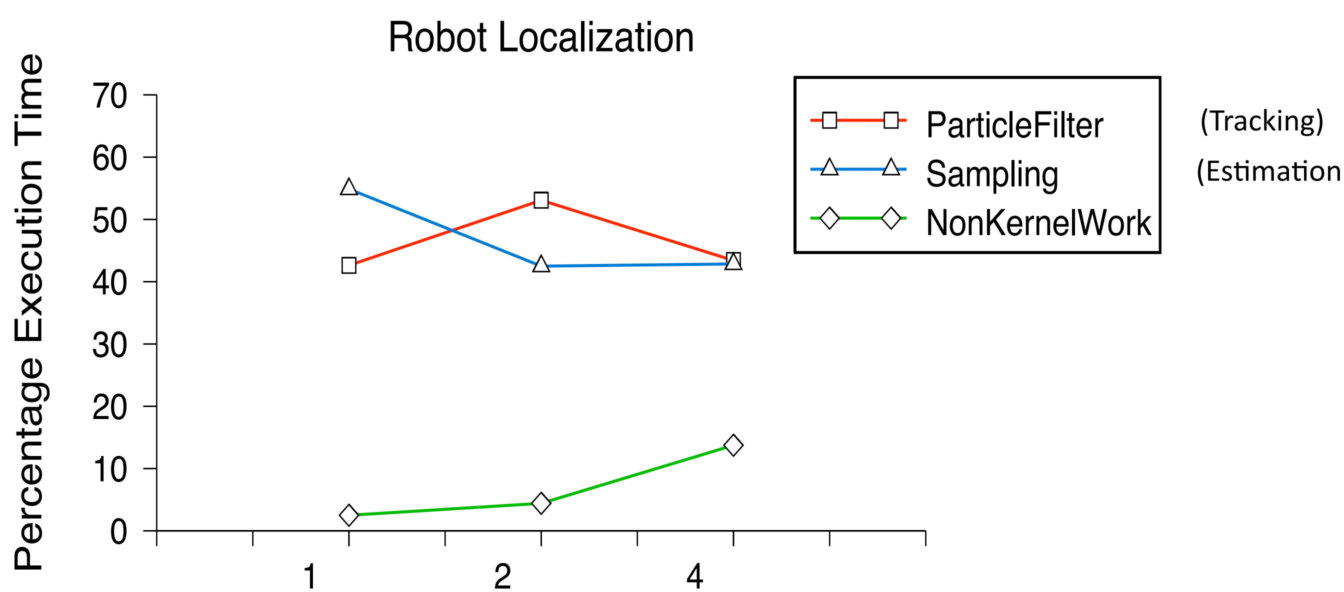
Robot Localization: Overview

Process of evaluating path based
on obstacles and a set of goals



Applications
- Space exploration programs
- Autonomous vehicles/systems
- Mobile robotics

Robot Localization: Algorithm



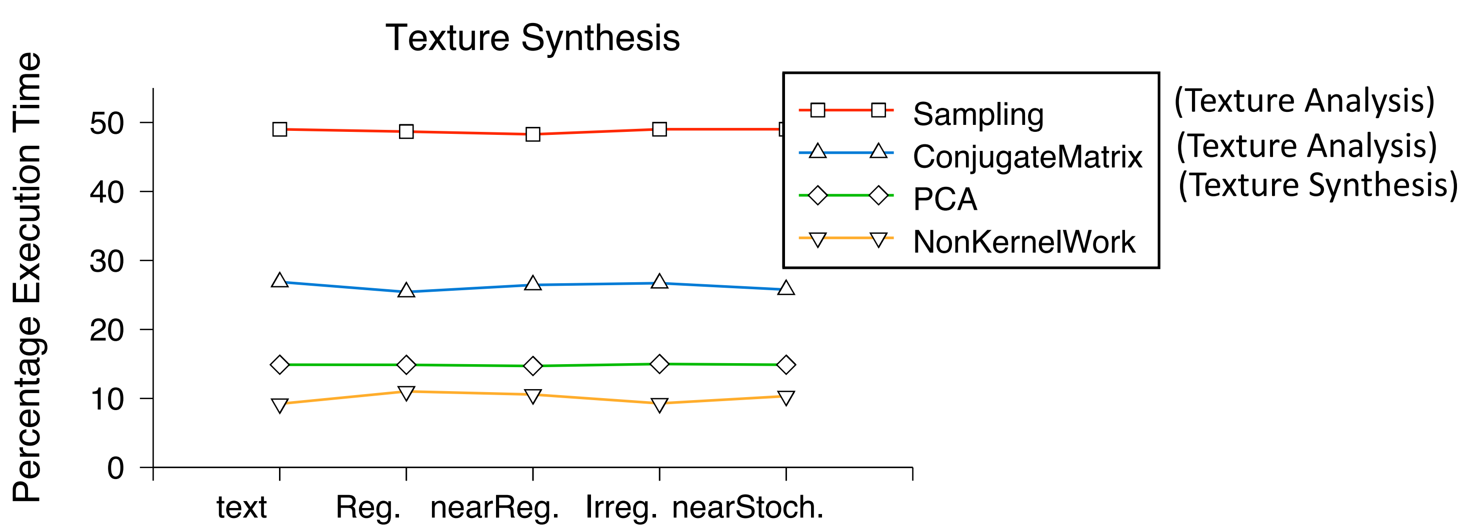
Texture Synthesis: Overview

Constructing large images from small image using structural content

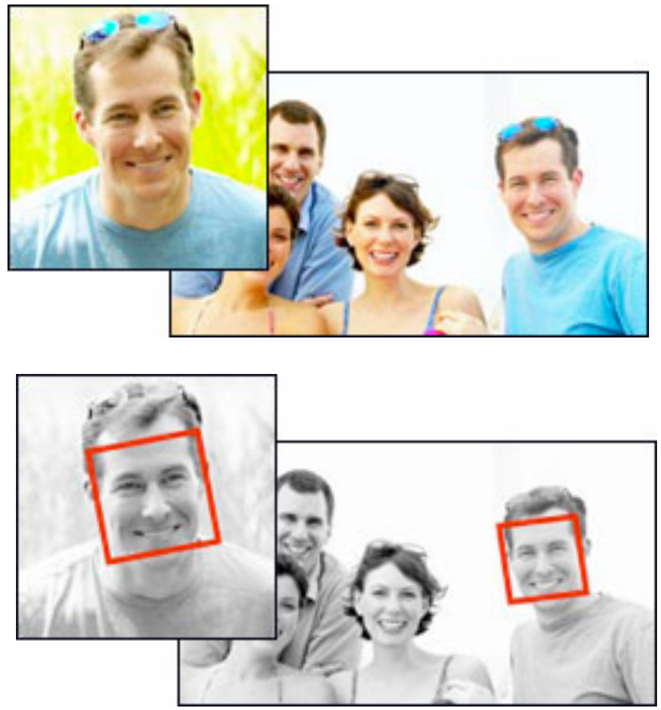
Applications

- Movie making
- Graphics
- Computational Photography

Texture Synthesis: Algorithm



Face Detection: Overview



Determines locations and sizes of human faces

Applications

- Object tracking
- Digital Camera, Facebook
- Content based image retrieval

Results: Execution Times

These times are for 1 or 2 frames of computation on a single core.

Lots of potential for making use of multi-core processors! Far enough away that we can use lots of cores, but close enough that it's attainable with more cores.

Benchmark	12k pixels	25k pixels	100k pixels
Feature tracking	2.77	5.0	19.4
Disparity Map	0.8	1.8	6.2
Image Stitch	0.7	10.1	23.4
SIFT	17.4	44.5	131
Texture Synthesis	18.5		
Image Segmentation	8.3	9.2	8.4
SVM	Up to Thousands		

Seconds per Frame

Results:
Parallelism in Vision Kernels

Parallelism calculation:

- We implemented a transformation pass in LLVM infrastructure track operands through the program and determine the longest dependence chain through memory, control and instruction dependences.
- Parallelism = # Instrs / Critical Path Length

The benchmarks shown on the right have lots of parallelism!

(If you are curious, we encourage you to go ahead and download our benchmark suite!)

Benchmark	Kernel	Approx. Parallelism
Disparity	Correlation	502
	Integral Image	160
	Sort	1,700
	SSD	1,800
Feature Tracking	Gradient	71
	Gaussian Filter	637
	Integral Image	1,050
	Area Sum	425
	Matrix Inversion	171,000
SIFT	SIFT	180
	Interpolation	502
	Integral Image	16,000
Image Stitch	LS Solver	20,900
	SVD	12,300
	Convolution	4,500
SVM	Matrix Ops	1,000
	Learning	851
	Conjugate Matrix	502

Related Work: Other vision codes

- Performance-oriented benchmarks
  - PARSEC: Body Tracking
  - MediaBench – image/video compression algorithms
  - Spec2000 facerec
- Accuracy-oriented benchmarks (for vision research)
  - Berkeley Segmentation Database and Benchmark
  - PEIPA
  - Muscle
  - ImageCLEF
- Vision Libraries
  - OpenCV- Highly tuned vision toolbox

Related Work: Intel OpenCV vs SD-VBS

	OpenCV	SD-VBS
Goal	Tuned, Feature-loaded implementations for commercial and academic vision applications	"Pure" versions for easy analysis, transformation and parallelization in multi-core architecture research
Source Code	C++	C or MATLAB, your choice
Platform specific optimizations	Highly Tuned for best performance on Intel and supporting architectures	Actively removed optimizations that increase code complexity
Coding style	Highly flexible; full of options to change behavior of each function	Clean code without features that deter analysis
Example: FILTER	2000 lines of code	50 lines of code
	204 conditional statements	No conditional statements
	Pointer Operations	Array Operations
Ease of analysis	Harder	Easier, because simpler implementations

Conclusions

- Computer Vision is an exciting domain with immense potential
- Vision algorithms are full of parallelism, and can benefit from processors with greater and greater performance; which make them ideal for multi-core
- SD-VBS is a comprehensive and clean benchmark suite for vision, well suited for multi-core and many-core research.

Public release

http://parallel.ucsd.edu/vision

SD-VBS:
The San Diego Vision Benchmark Suite

http://parallel.ucsd.edu/vision

We would like to thank the following Vision researchers:

- Andrea Vedaldi
- Athanasois Noulas
- Tom Mitchell
- Javier Portilla
- Eero Simoncelli
- Jianbo Shi

For contributing their code and data sets to the benchmark suite.

Supported by NSF CAREER Award 0846152

* * * * *