

MATLAB Image Processing

By Bruce Tannenbaum
IEEE Boston Meeting, Dec 3, 2012

© 2012 The MathWorks, Inc.

Who uses MATLAB?



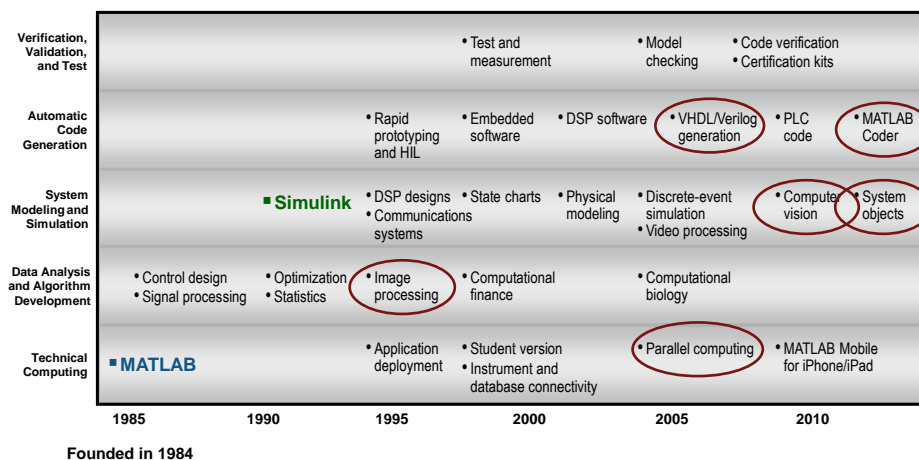
Key Industries

- Aerospace and defense
- Automotive
- Biotech and pharmaceutical
- Communications
- Computers
- Education
- Electronics and semiconductors
- Energy production
- Industrial automation and machinery
- Medical devices



3

Key Capabilities Drive MathWorks Business



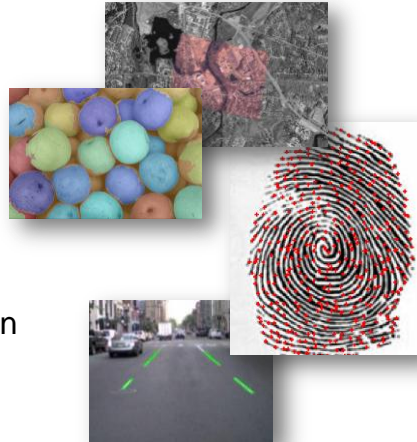
4



Image Processing Toolbox

Perform image processing, analysis, visualization, and algorithm development

- Image analysis
- Image enhancement
- Spatial transformation
- Image registration
- Morphological operations
- ROI-based processing
- Image display and exploration



5



Imaging-related MATLAB Toolboxes

- Image Acquisition Toolbox
- Image Processing Toolbox

- Computer Vision System Toolbox
 - Also supports video processing
- Parallel Computing Toolbox
 - For multicore computers, GPUs, and computer clusters
- Statistics Toolbox
 - Clustering, statistics, and machine learning

6

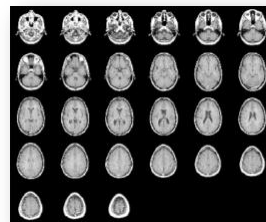
Agenda

- Working with large data sets
 - Block processing
 - Parallel processing
- Image registration
 - Intensity-based
 - Feature-based
- Video processing

7

Examples of Large Image Data Sets

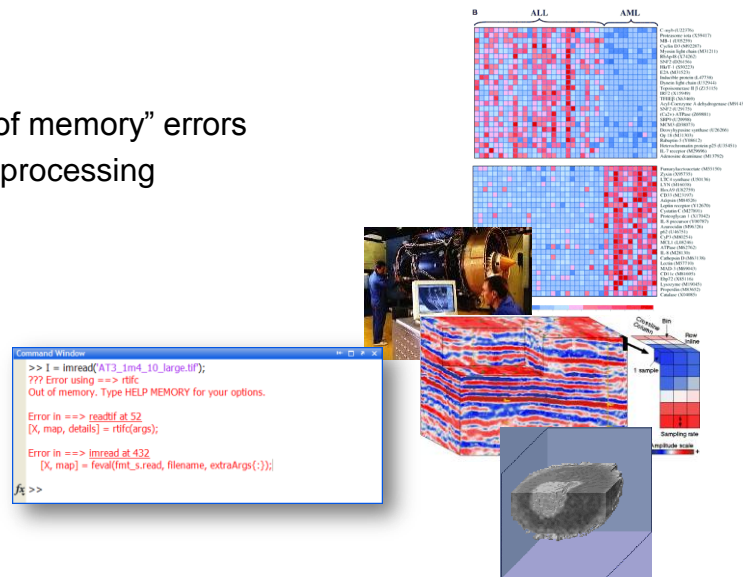
- Satellite imagery
- Aerial surveys
- Super-resolution
- Image sequences and stacks
- Volumetric data
- Multispectral and hyperspectral
- Mosaics and panoramic imagery



8

Common Large Image Challenges

- “Out of memory” errors
- Slow processing



9

blockproc and rsetwrite

- Block processing
 - Automatically divides an image into blocks for processing
 - Reduces memory usage
 - Processes arbitrarily large images
- Reduced resolution data set
 - Avoids memory demands in visualizing large images



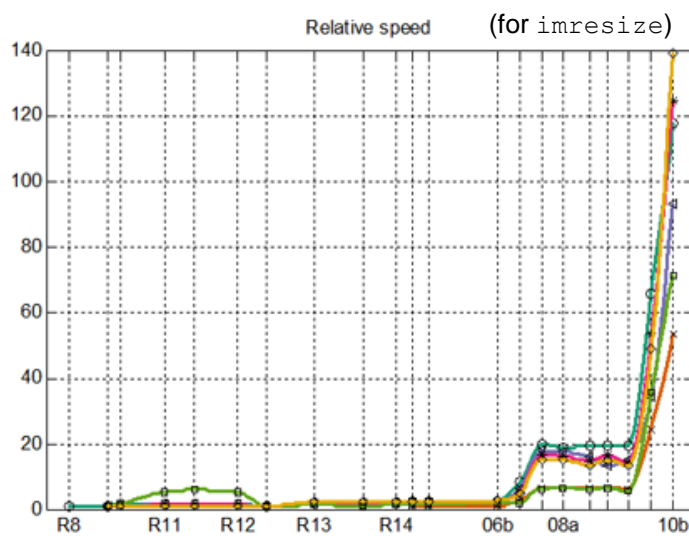
10

Improving Performance

- MATLAB
 - Preallocate space for variables
 - Identify bottlenecks with Profiler
 - Vectorize code
- Image Processing Toolbox
 - Many toolbox functions are faster as of R2010a
 - Implicit use of multicore processors
- Parallel Computing Toolbox
 - Explicitly use multicore processors, clusters, and GPUs

11

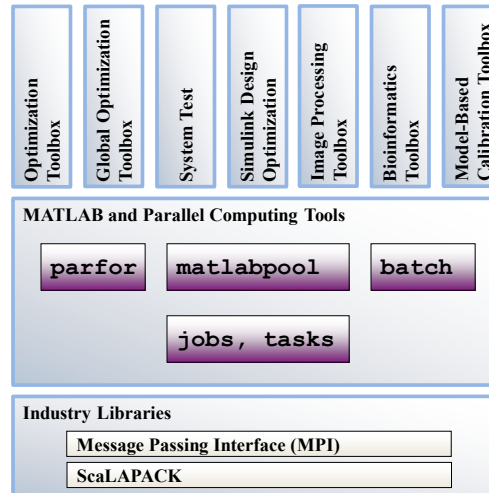
Example of IPT Performance Improvements



12

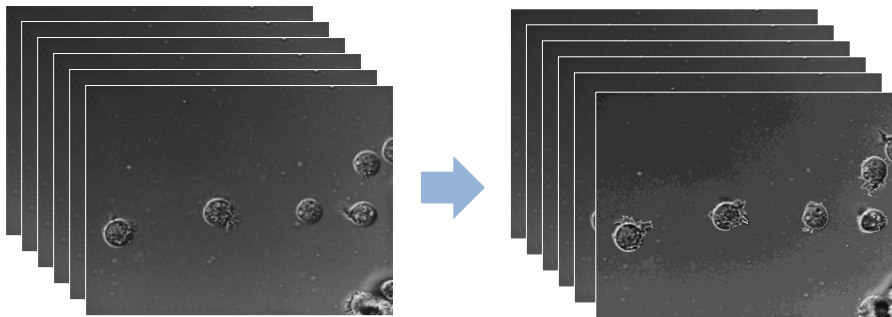
Parallel Computing with MATLAB

- Built in parallel functionality within specific toolboxes
- High level parallel functions
- Low level parallel functions
- Built on industry standard libraries



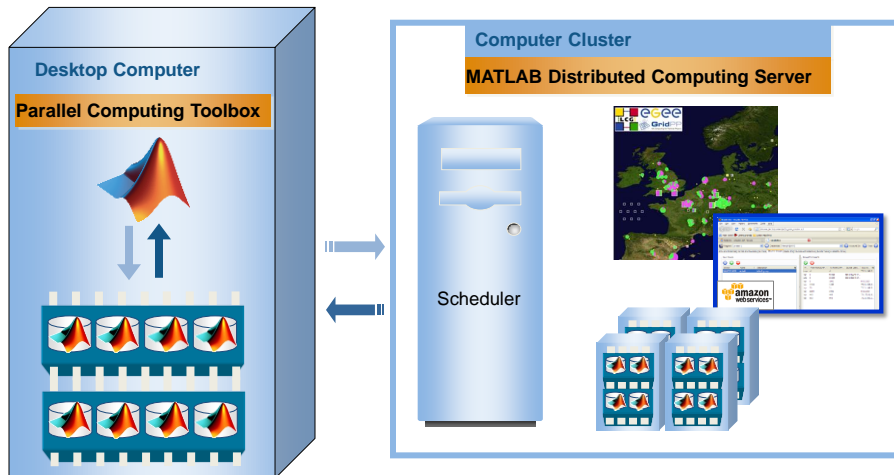
13

Parallelized Image Segmentation with `parfor`



14

Scale Up to Clusters, Grids and Clouds



15

GPU Support with Parallel Computing Toolbox

- Run supported MATLAB code on the GPU
 - Includes functions such as `conv2`, `filter2`
- Create kernels from existing CUDA code and PTX files
- Run kernels on the GPU from MATLAB



16

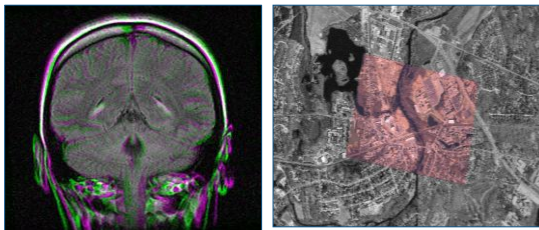
Agenda

- ✓ Working with large data sets
 - ✓ Block processing
 - ✓ Parallel processing
- Image registration
 - Intensity-based
 - Feature-based
- Video processing

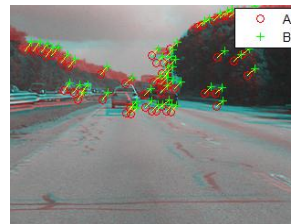
17

Automatic Image Registration

Intensity-based



Feature-based

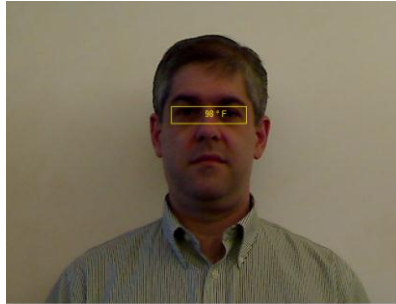


Two new registration types supported:

- Intensity-based (Image Processing Toolbox)
- Feature-based (Computer Vision System Toolbox)

18

Demo: Intensity-based Image Registration

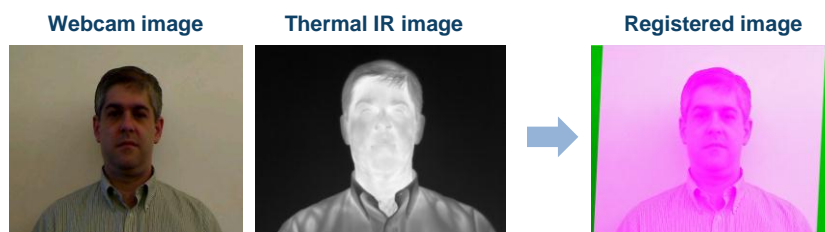


Goal:

- Detect febrile symptoms off an image

19

Demo: Intensity-based Image Registration



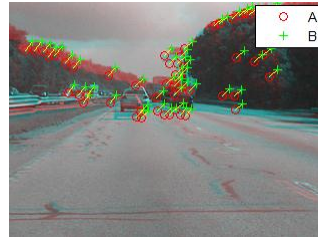
Step 1: Register the image pair to detect the eyes

Step 2: Read temperature near eyes & issue alert in case of fever

20

Feature-based Registration

- Aligning data from multiple sources
- Applications
 - Stabilization
 - Mosaicking, panoramas
 - Stereo vision
 - Data fusion
- Key capabilities
 - Feature detection and extraction
 - Feature matching
 - RANSAC for estimating geometric transformation or fundamental matrix



21

Demo: Feature-based Image Registration

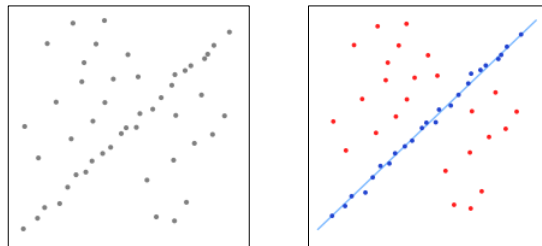
- Workflow
 - Feature detection and extraction
 - Feature matching
 - Geometric transformation estimation with RANSAC



22

RANSAC for Estimation

- Random Sample Consensus
 - Iterative estimation of parameters to a mathematical model from a set of observed data that contains outliers
- Our uses of RANSAC
 - Estimate Geometric Transformation
 - Estimate Fundamental Matrix (useful for stereo vision)



23

Agenda

- ✓ Working with large data sets
 - ✓ Block processing
 - ✓ Parallel processing
- ✓ Image registration
 - ✓ Intensity-based
 - ✓ Feature-based
- Video processing

24



Video Processing in MATLAB

```
myVid = VideoReader('myvideofile.avi');
numFrames = myVid.NumberOfFrames;
numIter = 10;
opticalFlowIn = zeros([size(currentFrame) 5]);
opticalFlowOutput = zeros([size(currentFrame) numFrames]);

i = 1;
while i <= numFrames
    opticalFlowIn(:, :, 2:end) = opticalFlowIn(:, :, 1:end-1);
    opticalFlowIn(:, :, 1) = read(myVid, i);

    flow = opticalFlow(opticalFlowIn(:, :, 1), opticalFlowIn(:, :, 5), ...
        'horn-schunck', numIter, 'magitude-squared');

    opticalFlowOutput(:, :, i) = flow;

    i = i+1;
end

implay(opticalFlowOutput, 30)
```

Explicit state management

Explicit indexing

Need to maintain buffer

25



Video Processing with System Objects

```
reader = vision.VideoFileReader
reader.Filename = 'myvideofile.avi';
viewer = vision.DeployableVideoPlayer('framerate', 30);

optical = vision.OpticalFlow
optical.Method = 'horn-schunck';
optical.OutputValue = 'Magitude-squared';
optical.ReferenceFrameDelay = 3;
optical.MaximumIterationCount = 10;

while ~isDone(reader)
    currentFrame = step(reader);
    OF = step(optical, currentFrame);
    step(viewer, OF);
end
```

Initialize objects

"In-the-loop" code is much simpler

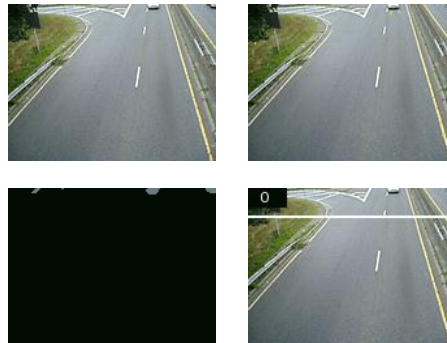
Implicit states, buffering, and indexing

Video player runs in-the-loop

26

Demo: Car Detection Using Optical Flow

Use optical flow to detect and count moving vehicles on a road



27

Image Acquisition Toolbox Hardware Support

- Manufacturers include:
 - Allied Vision Technologies
 - Basler
 - Baumer
 - DALSA
 - FLIR
 - Hamamatsu
 - Lumenera
 - Matrox Imaging
 - National Instruments
 - PixeLINK
 - Point Grey
 - Qimaging
 - Sony
 - And many more



HAMAMATSU

- See Supported Hardware Pages for more information

28

Why use MATLAB for Image and Video Processing?


- Read and write many image file formats
- Visualize and explore images interactively
- Connect directly to cameras and frame grabbers
- Use a large library of built-in functions
- Quickly build custom image processing algorithms
- Block-process large images to avoid memory issues
- Process images faster with multiple cores and clusters

29

Additional Resources

MATLAB Central Blog: “Steve on Image Processing”

<http://blogs.mathworks.com/steve/>



An open exchange for the MATLAB and Simulink user community

Hosted by The MathWorks

Search

[File Exchange](#)
[MATLAB Newsgroup](#)
[Link Exchange](#)
[Blogs](#)
[Contest](#)
[MathWorks.com](#)

Steve on Image Processing

October 2nd, 2007

Upslope area - influence and dependence maps

In my [August 7th post](#) on upslope area, I showed how to construct and solve the flow matrix to determine the upslope area for every pixel in a digital elevation model (DEM). In addition, the flow matrix can be used to compute both the *influence map* and the *dependence map*.

The influence map shows where water starting from a particular pixel will drain. The dependence map shows which uphill pixels drain through a particular pixel. Let's look at how to compute these from the flow matrix.

Recall the form of the flow equation for a particular pixel. The upslope area of a pixel equals 1 plus a weighted fraction of the upslope area of each of its neighbors.

$$A_j = 1 + w_1 A_{n_1} + w_2 A_{n_2} + \dots + w_9 A_{n_9}$$

The 1 term comes from assuming each pixel contributes an equal volume to the overall flow across the terrain. Think of it as rain falling equally on each pixel. To compute the influence map, then, we just have to modify our set of equations so that the 1 term appears only for the pixel or pixels of interest. Fortunately, that only affects the right-hand side of the equation, not the flow matrix itself. So solving for the influence map a particular pixel looks almost like solving for upslope area for all pixels.

Here's an example from the Milford, Massachusetts DEM that I've used before:


```

s = load('milford_ma_dem');
E = s./20;

```

About

Steve Eddins manages the Image & Geospatial development team at The MathWorks and coauthored [Digital Image Processing Using MATLAB](#). He writes [here](#) about image processing concepts, algorithm implementations, and MATLAB.

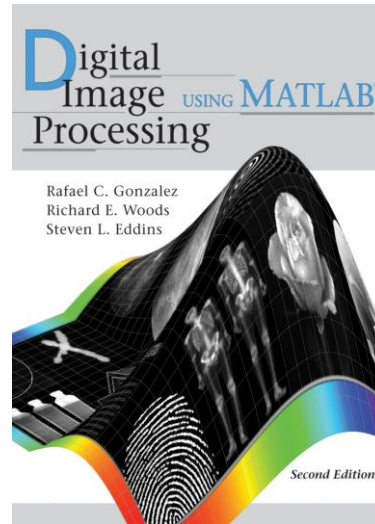


30

Additional Resources

Digital Image Processing Using MATLAB

Gonzalez, Woods, and Eddins
Gatesmark

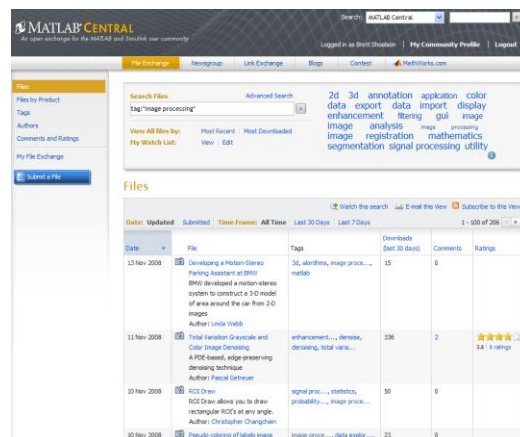


31

Additional Resources

MATLAB Central File Exchange

<http://www.mathworks.com/matlabcentral/fileexchange/>



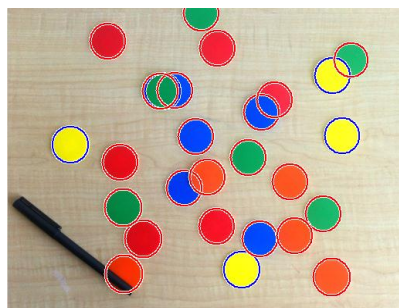
Date	File	Tags	Downloads (last 30 days)	Comments	Rating
13 Nov 2008	Developing a Motion-Denoise Filtering Algorithm at 30FPS (BRI) developed a motion-denoise system to construct a 3-D model of a car around the car flow 3-D images	3d, algorithms, image proc...	15	0	
11 Nov 2008	Total Variation-Guided and Color Image Denoising A-Prior based, edge-preserving denoising technique	enhancement, ..., denoise, denoising, total vari...	206	2	4.5 stars (14 ratings)
10 Nov 2008	ROI Draw ROI Draw allows you to draw rectangular ROIs at any angle.	signal proc..., statistics, probability..., image proc...	50	0	
10 Nov 2008	Pseudo coloring of false-color image	image proc..., data expl...	23	0	

32

Questions?

© 2012 The MathWorks, Inc.

If Time Permits.. Find the Circles



Goal:

- Find circles in truecolor image
- Draw edge lines around the circles