Machine Vision for Precision Agriculture

Matthew Dailey

Vision and Graphics Lab
Computer Science and Information Management
Asian Institute of Technology
Bangkok Thailand

References

Some references for these lecture notes:

- Bishop, C. *Pattern Recognition and Machine Learning*, Springer, 2007.
- Hartley, R., and Zisserman, A. *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2004.

Acknowledgments

The research presented here is joint work with several colleagues:

- Supawadee Chaivivatrakul, Ubon Ratchathani University
- Jednipat Moonrinta, AIT
- Atima Tharatipyakul, Singapore University of Technology and Design (SUTD)
- Waqar Qureshi, National University of Science and Technology, Pakistan
- Aayush Rana, AIT
- Lee Tang, Iowa State University
- Akash Dev Nakarmi, Iowa State University

Outline

1. Preliminaries
2. Introduction to 3D computer vision
3. Texture modeling with SIFT
4. Classification with SVMs
5. Agricultural crop mapping
6. High-throughput plant phenotyping
7. Conclusion

Some of the major phases of agricultural production:

- Crop cultivation
- Water management
- Fertilizer management
- Pest management
- Harvest
- Post harvest handling
- Transportation
- Food packaging
- Food processing and storage
- Food storage
- Food quality and safety management
- Marketing

ICT in general can help improve quality and productivity in all of these phases.

Today, we look at pre-harvest sensing techniques from machine vision useful for precision agriculture.

Topics to be covered today:

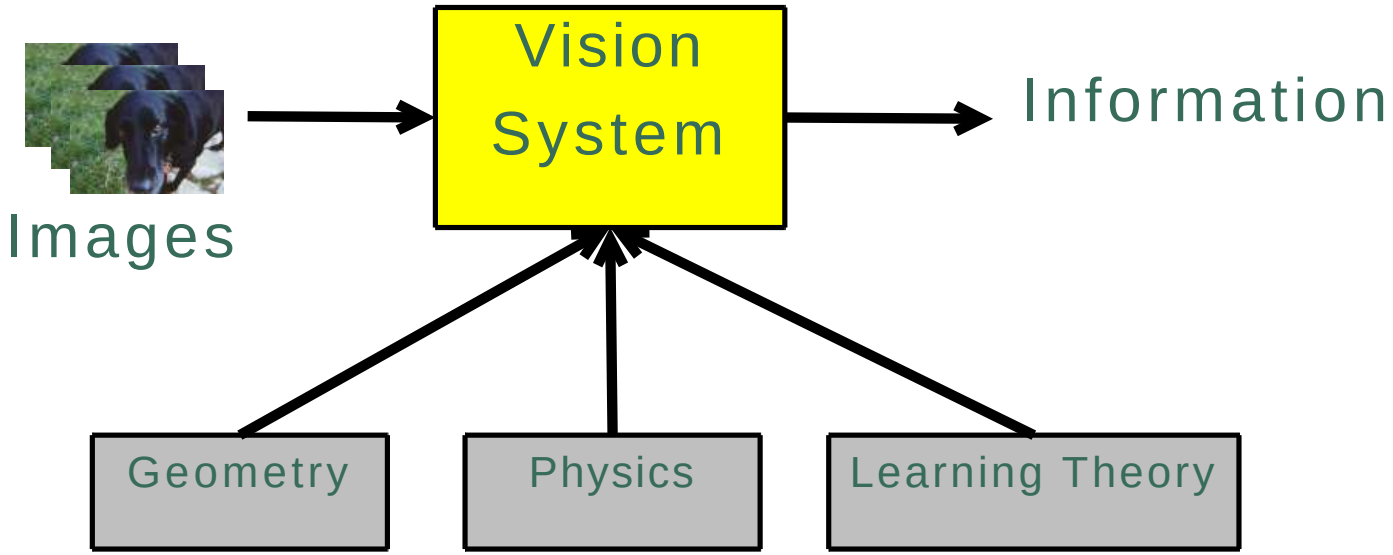
- Introduction to machine vision
- Applications in plant phenotyping
- Applications in crop mapping and density estimation
- Applications in yield estimation and prediction

Hands-on exercises in the afternoon workshop:

- Learning color histograms from data
- Backprojection of color histograms for object detection and density estimation
- Learning texture models
- Texture-based object detection and classification

Applying machine vision in practice requires a vision system.

First, what is a vision system?



The kind of information we want is application specific:

- 2D or 3D models
- Object categories or appearance
- Object poses
- Camera poses

Important applications of vision systems include:

- Mobile robot navigation
- Industrial inspection and control
- Military intelligence
- Security
- Human-computer interaction
- Image retrieval from digital libraries
- Medical image analysis
- 3D model capture for visualization and animation
- Agricultural monitoring

What kinds of information might we want to extract in agricultural monitoring applications?

Small exercise:

Each participant: think of one application of a vision system for phenotyping, density mapping, or yield estimation and how it could improve productivity.

[Briefly discuss the ideas.]

OK, we'll see if these ideas broaden after some of our lectures.

A "vision system" includes:

- Platform
  - Fixed platform (field sensor)
  - Mobile platform (ground robot, hand-held device, airborne vehicle)
- Image acquisition hardware
  - Digital camera with a fast serial interface (USB, etc.)
  - IP camera with WiFi or wired network interface
  - Self-contained DLSR or action camera
- Computing hardware
  - On board (embedded system, tablet, smartphone)
  - Remote (workstation, laptop, tablet, smartphone)
- Image processing support software
- Our computer vision algorithms

In crop monitoring, same as any IP application, we have

- 2D or 3D structure
- Appearance

Plant and other objects' structure can change slowly (growth) or quickly (wind effects, people, animals, or machines moving in the scene).

Intrinsic appearance changes slowly but extrinsic appearance can change quickly due to lighting changes.

Extracting structure and intrinsic appearance are the two key problems for image processing.

Now we have a brief introduction to the computer vision technology needed for precision agriculture applications.

To measure structure and appearance parameters needed in agricultural monitoring applications, there are three concerns:

- 3D geometry and structure from motion
- Color modeling
- Texture modeling

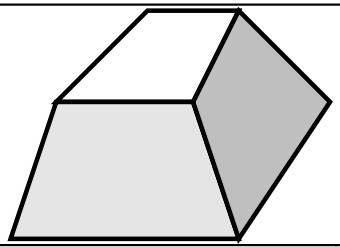
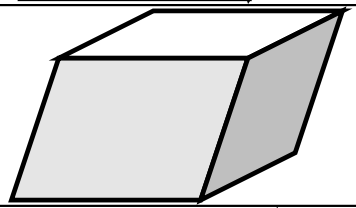
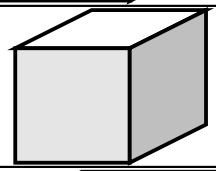
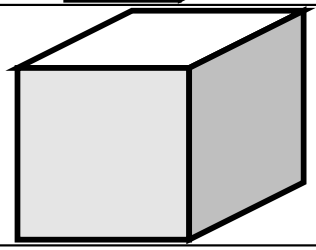
We begin with 3D geometry and structure from motion, which is similar to photogrammetry in remote sensing.

Then we will introduce color modeling and texture modeling in the context of precision agriculture applications.

Outline

1. Preliminaries
2. Introduction to 3D computer vision
3. Texture modeling with SIFT
4. Classification with SVMs
5. Agricultural crop mapping
6. High-throughput plant phenotyping
7. Conclusion

3D objects imaged in 2D can often be recovered up to some projective transformation:

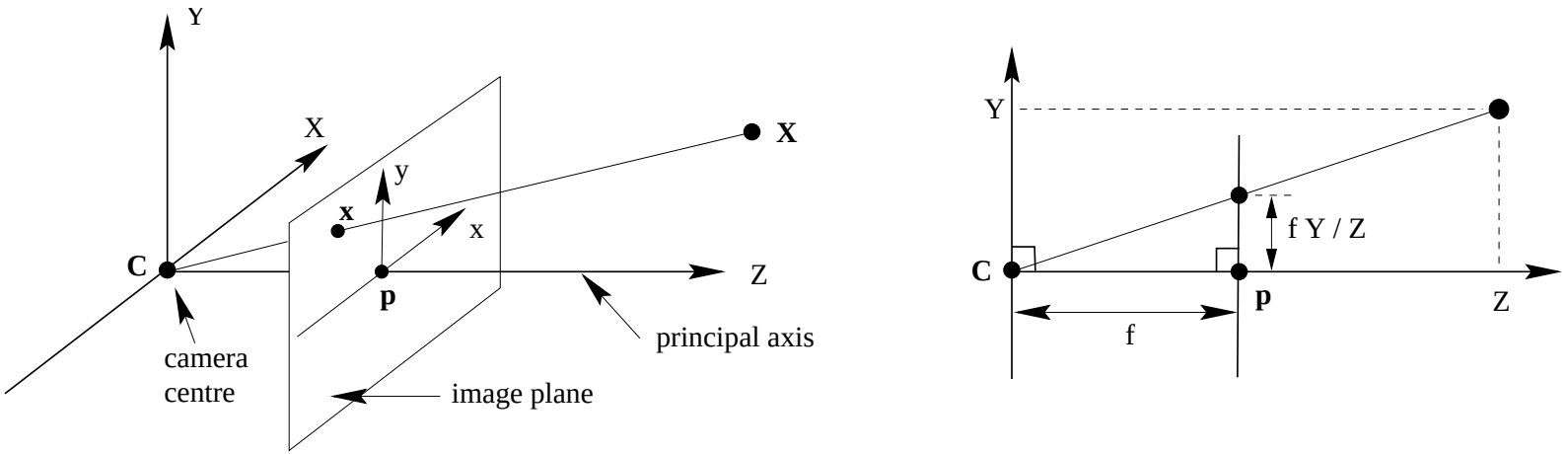
Group	Matrix	Distortion	Invariant properties
Projective 15 dof	$\begin{bmatrix} \mathtt{A} & \boldsymbol{t} \\ \boldsymbol{v}^T & v \end{bmatrix}$		Intersection and tangency of surfaces in contact. Sign of Gaussian curvature.
Affine 12 dof	$\begin{bmatrix} \mathtt{A} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix}$		Parallelism of planes, volume ratios, centroids. The plane at inifinty $\boldsymbol{\pi}_\infty$
Similarity 7 dof	$\begin{bmatrix} s\mathtt{R} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix}$		The absolute conic Ω_∞.
Euclidean 6 dof	$\begin{bmatrix} \mathtt{R} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix}$		Volume

Hartley and Zisserman (2004), Table 3.1

The pinhole camera uses central projection of points onto a plane.

The camera center or optical center, is the center of projection and the origin of a Euclidean coordinate system.

The image plane or focal plane is the plane $Z = f$.



Hartley and Zisserman (2004), Fig. 6.1

Exercise: If an object known to be 180cm high is 90 pixels tall in an image from a camera with focal length of 540 pixels, how far away is the object from the camera along the Z axis?

The transformation from a point in 3-space is just

$$(X, Y, Z)^T \mapsto (fX/Z, fY/Z)^T$$

In homogeneous coordinates, this is a linear transformation:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}.$$

We write this compactly as

$$\boldsymbol{x} = \mathtt{P}\boldsymbol{X}$$

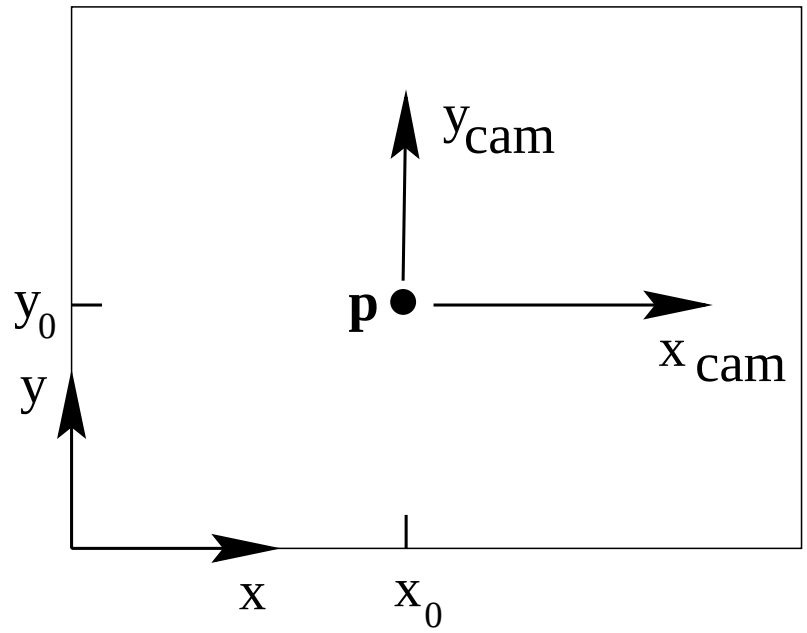
where

$$\mathtt{P} = \mathrm{diag}(f, f, 1) \left[\mathtt{I} \mid \mathbf{0} \right].$$

If the coordinate system in the image plane is not centered at the principal point, we write

$$(X, Y, Z)^T \mapsto (fX/Z + p_x, fY/Z + p_y)^T$$

where $(p_x, p_y)^T$ are the coordinates of the principal point.



Hartley and Zisserman (2004), Fig. 6.2

Now we write the transformation as

$$\boldsymbol{x} = \mathtt{K} \left[\mathtt{I} \mid \mathbf{0} \right] \boldsymbol{X}_{\text{cam}}.$$

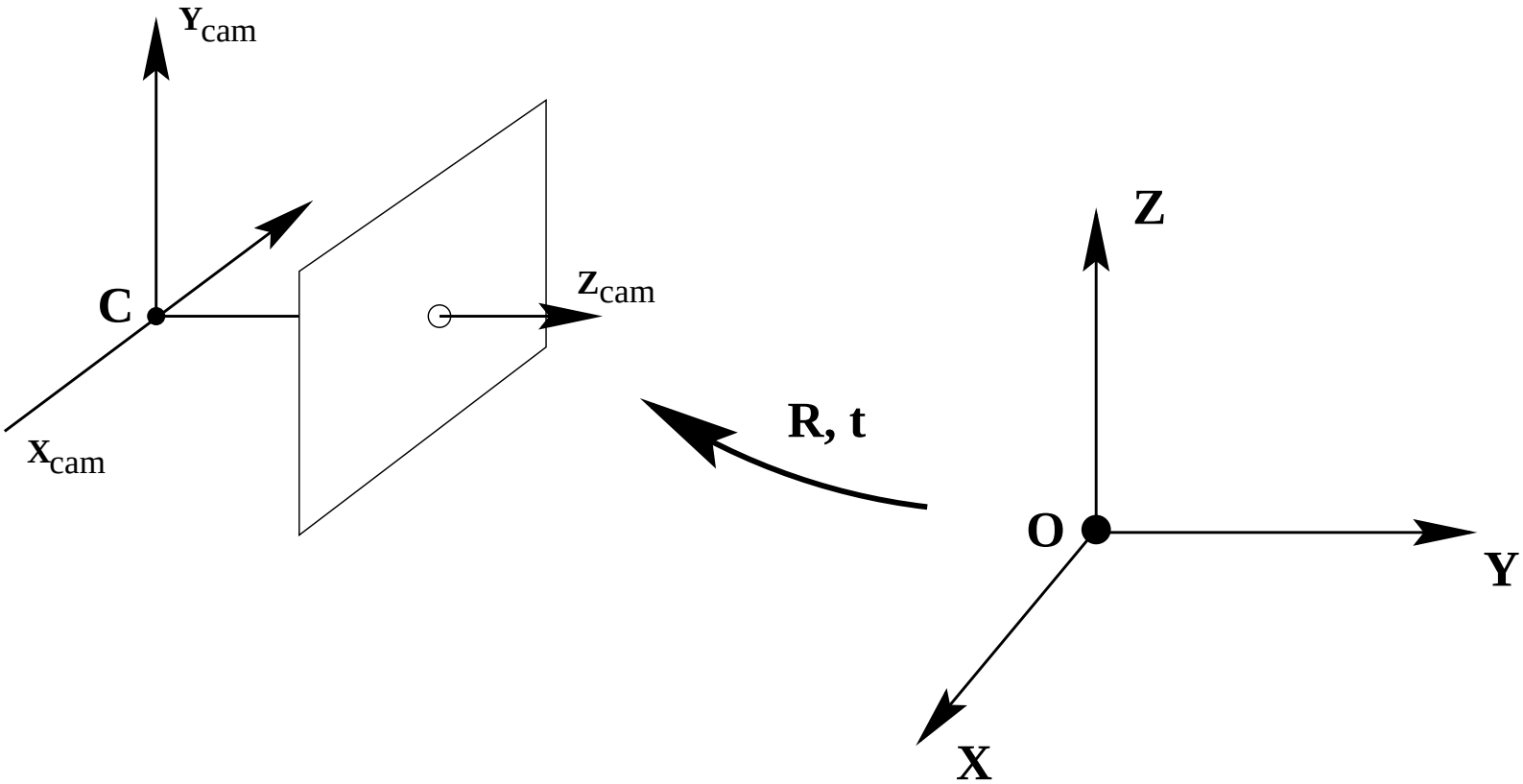
where $\mathtt{K}$, called the camera calibration matrix is

$$\mathtt{K} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

The notation $\boldsymbol{X}_{\text{cam}}$ emphasizes that $\boldsymbol{X}$ is a 3D point in the camera coordinate frame.

The camera coordinate frame is a coordinate system whose origin is at the camera center and whose Z axis is the principal axis of the camera.

Now suppose our camera is rotated and translated with respect to a world coordinate frame:



Hartley and Zisserman (2004), Fig. 5.3

If the 3D point $\tilde{\boldsymbol{C}}$ is the camera center in the world coordinate frame, we write

$$\boldsymbol{X}_{\text{cam}} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\tilde{\boldsymbol{C}} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\tilde{\boldsymbol{C}} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \boldsymbol{X}.$$

Putting the rigid transformation together with the camera projection gives us

$$\boldsymbol{x} = \mathtt{K}\mathtt{R}\left[\mathtt{I} \mid -\tilde{\boldsymbol{C}}\right] \boldsymbol{X}$$

We write the general pinhole camera

$$\mathtt{P} = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\tilde{\boldsymbol{C}}],$$

a matrix with 9 degrees of freedom (6 for the rigid transform, two for the principal point, and 1 for the focal length f).

The matrix $\mathtt{K}$ is said to contain the intrinsic parameters of the camera.

$\mathtt{R}$ and $\tilde{\boldsymbol{C}}$ are the extrinsic parameters of the camera.

Usually we don't bother to make the camera center explicit and write

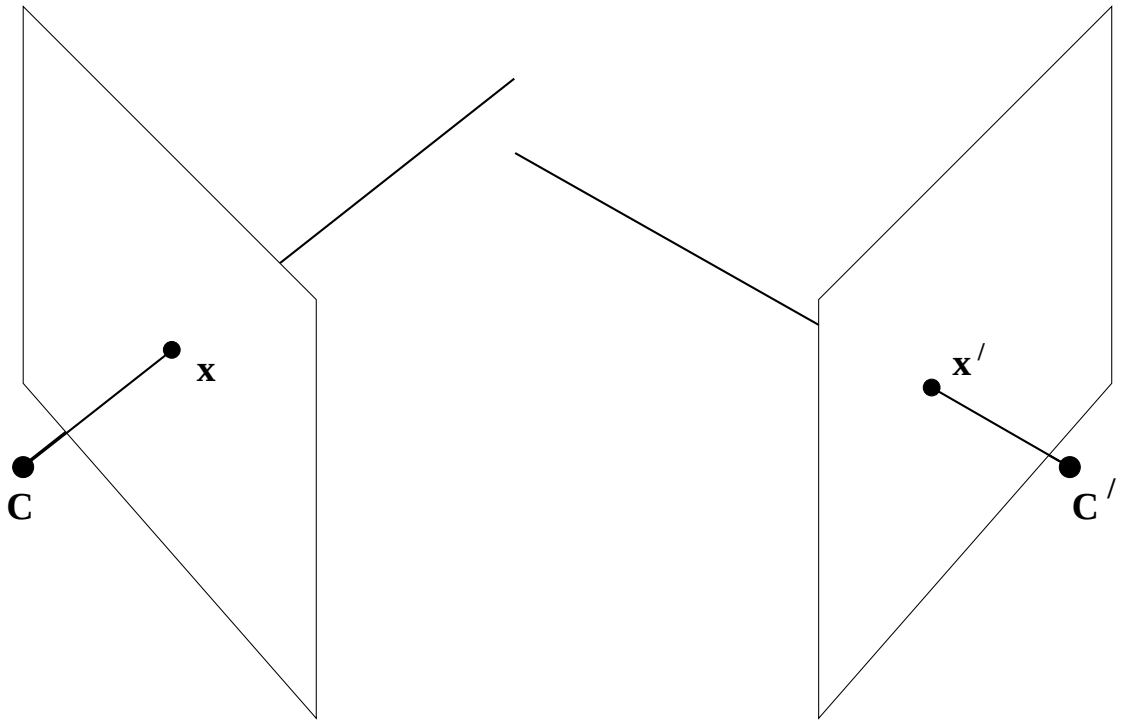
$$\mathtt{P} = \mathtt{K}[\mathtt{R} \mid \boldsymbol{t}]$$

where $\boldsymbol{t} = -\mathtt{R}\tilde{\boldsymbol{C}}$.

Finding $\mathtt{P}$ is one of the central problems of 3D vision.

How can we compute the position of a point in 3 space given two views and a perfect estimate of $\mathtt{P}$ and $\mathtt{P}'$, up to some ambiguity?

Backprojection for two corresponding points $\boldsymbol{x} \leftrightarrow \boldsymbol{x}'$ doesn't work because with image measurement error, the backprojected rays will be skew:



Hartley and Zisserman (2004), Fig. 12.1a

Despite the problem of skew, there are several methods for triangulation given known cameras and correspondences.

So if we know $\mathtt{P}$ and $\mathtt{P}'$ for two images and have corresponding 2D points $\boldsymbol{x}$ and $\boldsymbol{x}'$ for the two images, we can triangulate to find $\boldsymbol{X}$, the corresponding 3D point.

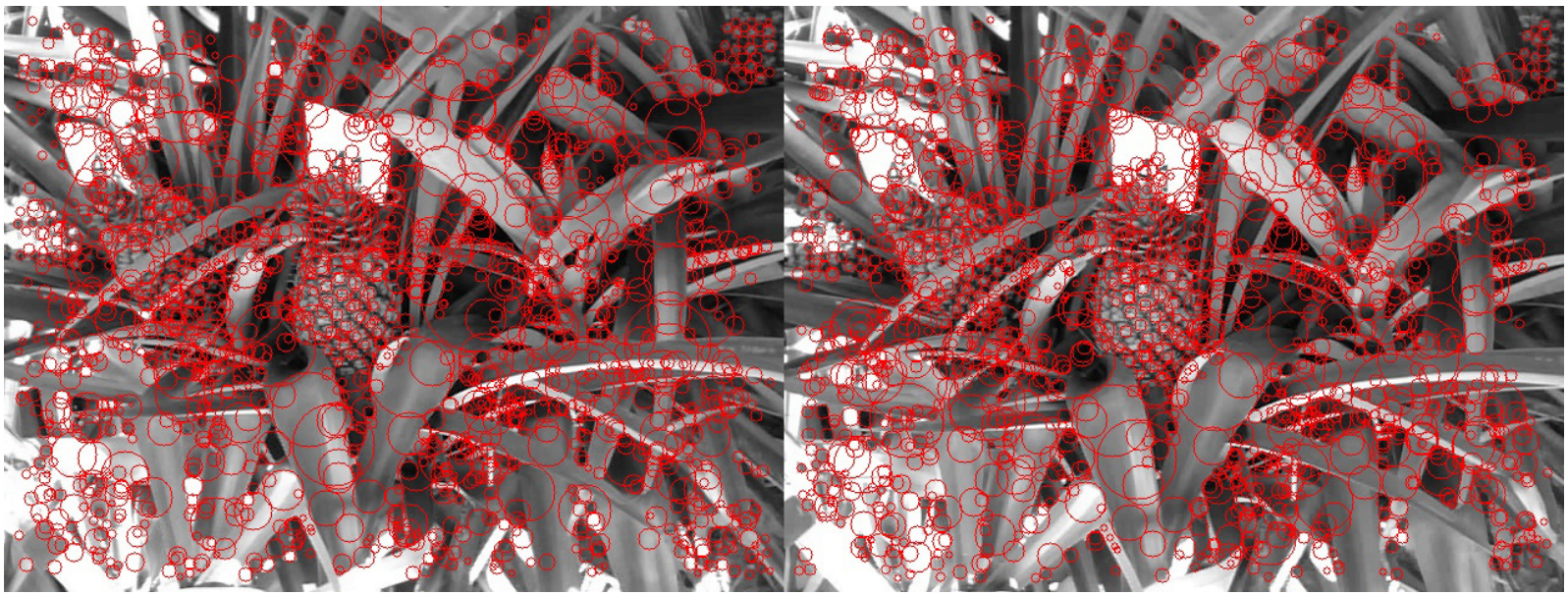
OK. Triangulation requires $\mathtt{P}$, $\mathtt{P}'$, $\boldsymbol{x}$, and $\boldsymbol{x}'$.

To find camera matrices $\mathtt{P}$, $\mathtt{P}'$, we first find corresponding points (hard), compute the epipolar geometry describing the relationsip between the two images (easy), then, if we know $\mathtt{K}$, $\mathtt{K}'$ we have the camera matrices.

Remaining: how to find corresponding points?

How to find corresponding points $\boldsymbol{x}$ and $\boldsymbol{x}'$ in two images?

- We can track points over time using optical flow techniques.
- We can search for correspondences using an interest point operator and match interest point descriptors. Algorithms include SIFT, SURF, ASIFT, ORB, etc.



Example SIFT interest points for two frames of a field video.

Matching of SIFT interest points for two frames of a field video.

Methods for tracking and matching are constantly being refined and improved.

Putting these elements together, we have structure from motion algorithms. Given a video sequence:

1. Extract a sequence of keyframes from the video.
2. Find correspondences between successive keyframes using tracking or feature point matching.
3. Calculate relative camera motion between keyframes
4. Triangulate 2D correspondences
5. Refine the solution using bundle adjustment (nonlinear least squares optimization of camera matrices and 3D points).

Today, with optimized methods such as those used in ORB-SLAM, tracking, matching, and structure computation can be done in real time with commodity hardware.

Outline

1. Preliminaries
2. Introduction to 3D computer vision
3. Texture modeling with SIFT
4. Classification with SVMs
5. Agricultural crop mapping
6. High-throughput plant phenotyping
7. Conclusion

Texture modeling with SIFT

Introduction

We just saw how 3D vision and structure from motion requires a set of point correspondences $\boldsymbol{x}_i \leftrightarrow \boldsymbol{x}'_i$.

When there is significant motion between two images, a rich feature detector with invariance to rotation and scale is desirable.

Besides 3D structure, to identify objects in agricultural applications, we need to extract information about texture (the structure of the intensity gradients in an image region).

Texture modeling with SIFT

Introduction

Lowe's (2004) Scale Invariant Feature Transform (SIFT) is an example of a method that can do both:

- It identifies feature points with interesting local gradient structure
- Given a feature point location, it computes a descriptor vector that characterizes the texture around the point
- The vector of texture attributes is somewhat invariant to scale and rotation in the plane and robust to moderate amounts of various kinds of affine distortion and noise.

Texture modeling with SIFT

SIFT steps

SIFT (Lowe, 2004) performs four basic steps:

- **Scale-space extrema detection:** A difference-of-Gaussian filter is run at several scales to find points that are local maxima or minima in space and scale.
- **Keypoint localization:** For each candidate location, the scale and location is determined, and unstable locations are discarded.
- **Orientation assignment:** Orientations are assigned to keypoint locations.
- **Keypoint descriptor:** A descriptor of the image intensities around the keypoint location is computed. The descriptors are obtained relative to the keypoint's location, orientation, and scale, so that the descriptor is invariant to scales, rotations (in the image plane), and translations.

The scale space is obtained by convolving Gaussians of various sizes with the input image:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

where $*$ is convolution and the 2D Gaussian is defined by

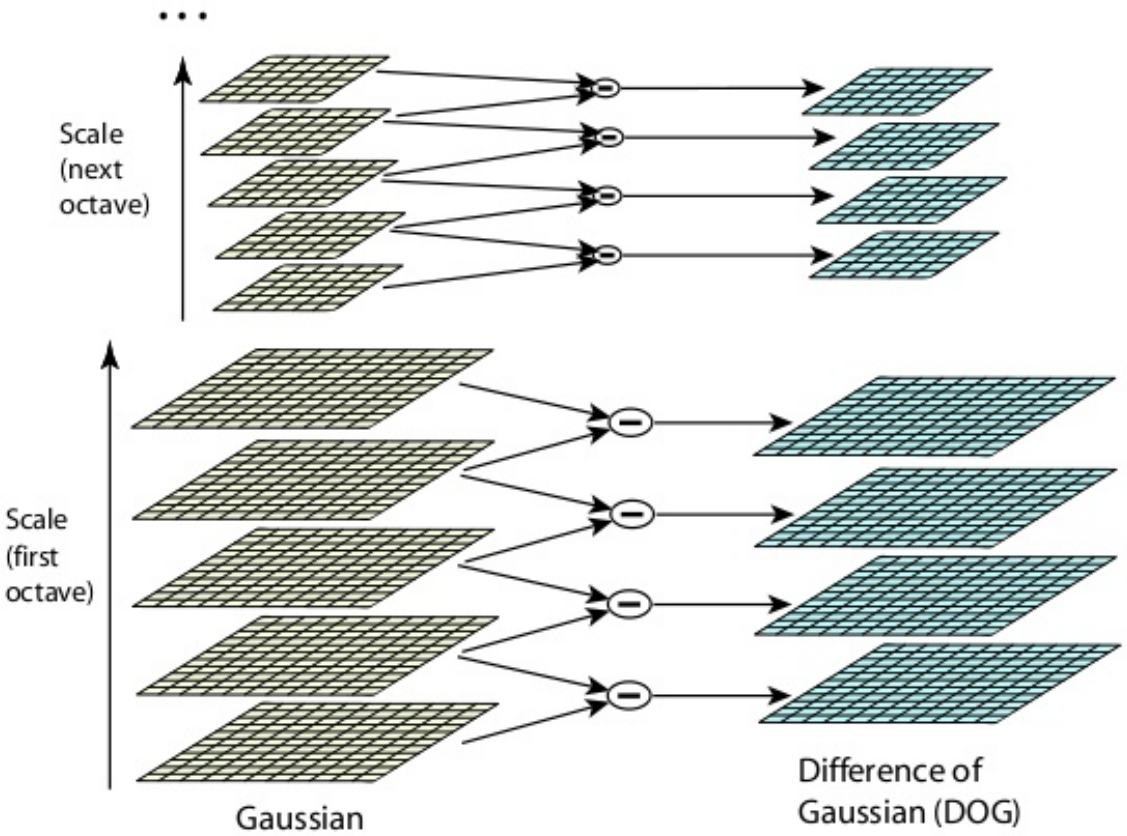
$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)}.$$

Stable keypoints are identified using the difference of Gaussian function:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned}$$

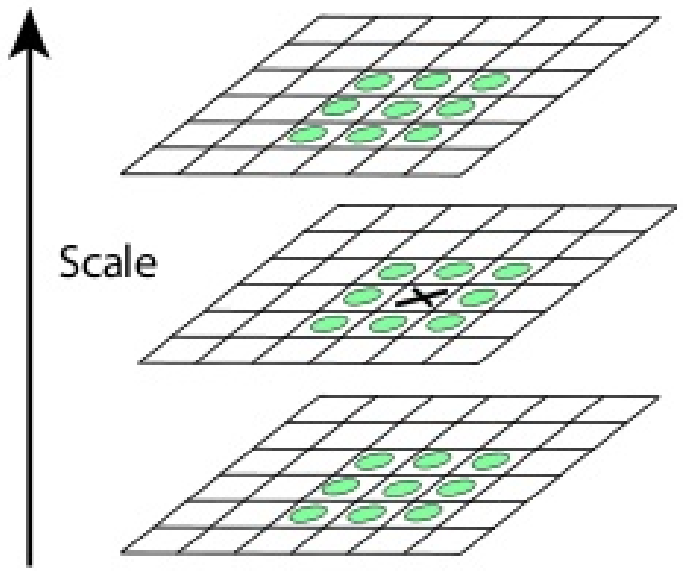
Texture modeling with SIFT

SIFT scale space



Lowe (2004), Fig. 1

Candidate keypoint locations are maxima and minima of the difference-of-Gaussian scale space in x, y, and scale:



Lowe (2004), Fig. 2

For the localization step, we could just take the location and scale at which the point was detected.

Instead, Lowe fits a quadratic function to the local values of $D(x, y, \sigma)$ then obtains a sub-pixel estimate of the location of the extremum, $\hat{\boldsymbol{x}}$.

Low-contrast extrema are immediately discarded.

Then, the eigenvalues of the 2×2 Hessian matrix around $\hat{\boldsymbol{x}}$ are examined to determine whether the extremum reflects a simple edge or a more complex corner-like region.

Simple edge-like candidate keypoints are discarded.

Using the Gaussian smoothed image at the scale closest to the detected difference of Gaussian extremum, we collect a histogram of image gradients in 36 directions in the region around the point.

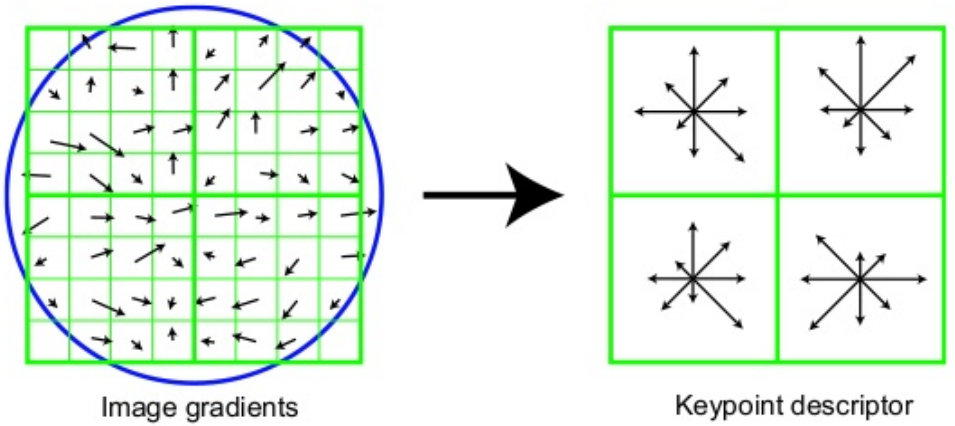
For the highest peak in the orientation histogram and any other strong peaks in the orientation histogram, SIFT creates a keypoint descriptor with that orientation.

This means we can get multiple keypoints for the same location (x, y, σ), but this only happens approximately 15% of the time.

Texture modeling with SIFT

SIFT keypoint descriptor

Finally, we compute a set of local histograms of gradient directions, relative to the dominant orientation.



Lowe (2004), Fig. 7

This is a 2x2 grid from a 16×16 sample array, but the standard implementations use 4×4 descriptor grid from a 16×16 sample array, to obtain a 128-element descriptor.

The blue circle is a Gaussian weighting window.

Texture modeling with SIFT

SIFT

Lowe's main goal was to perform object recognition in cluttered environments, but several computer vision and robotics groups have found SIFT useful for wide baseline matching to get correspondences for other algorithms.

We also find methods like SIFT useful for texture modeling in agricultural applications.

Outline

1. Preliminaries
2. Introduction to 3D computer vision
3. Texture modeling with SIFT
4. Classification with SVMs
5. Agricultural crop mapping
6. High-throughput plant phenotyping
7. Conclusion

Classification with SVMs

Introduction

We would like to use SIFT descriptors for texture modeling.

The most straightforward approach to texture-based object recognition would be to learn a function to classify the descriptor.

SIFT keypoints

Positively classified SIFT keypoints

We have a supervised binary classification scenario in which we have N points $\boldsymbol{x}_n \in \mathbb{R}^d$ with corresponding labels $t_n \in \{1, -1\}$.

Given the training dataset

$$\{(\boldsymbol{x}_1, t_1), \ldots, (\boldsymbol{x}_N, t_N)\}$$

we want to induce a decision rule predicting the sign of t for any point $\boldsymbol{x}$.

Over the last 20 years, the support vector machine (SVM) has emerged as one of the most reliable, effective, and easy to use binary classification techniques.

The SVM is a kernel method basing its prediction $y(\boldsymbol{x})$ on linear combinations of a kernel function $k(\cdot,\cdot)$ evaluated at the training data points:

$$k(\boldsymbol{x},\boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\phi}(\boldsymbol{x}')$$

where $\boldsymbol{\phi}(\boldsymbol{x})$ is a fixed transform into some feature space.

The simplest example of a kernel function is obtained when we use the identity map $\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}$ to obtain $k(\boldsymbol{x},\boldsymbol{x}') = \boldsymbol{x}^T\boldsymbol{x}'$, the so-called linear kernel.

Depending on the feature mapping $\phi(\boldsymbol{x})$, we can obtain many useful algorithms.

Rather than designing $\phi(\boldsymbol{x})$ directly, it is possible to write an arbitrary function for $k(\boldsymbol{x}, \boldsymbol{x}')$ then verify whether it is a valid kernel.

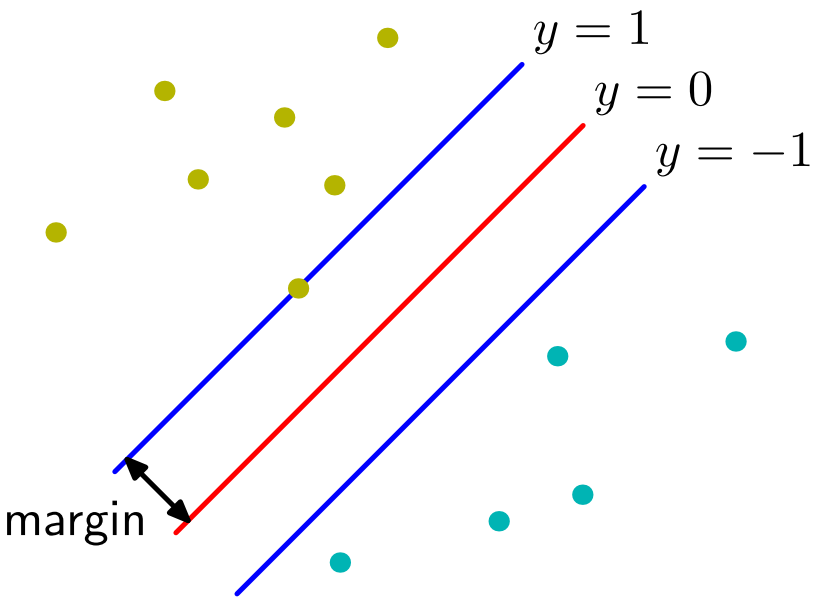
For the binary classification problem, we will estimate the parameter vector $\boldsymbol{w}$ in the linear model

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x}) + b.$$

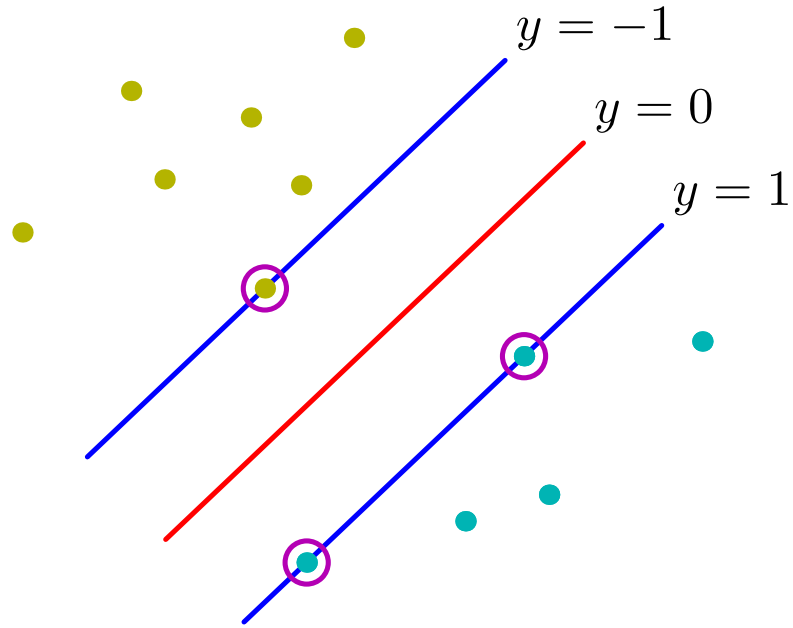
$\boldsymbol{w}$ defines a linear decision boundary (hyperplane) in feature space. Note that since $\phi(\boldsymbol{x})$ is nonlinear in general, the decision boundary will be nonlinear in the input space.

In the linearly separable case, the two classes -1 and 1 can be perfectly separated. Many hyperplanes will achieve 100% correctness, but we will choose the unique hyperplane maximizing the margin, or the smallest distance from the decision boundary to any training sample.

Margins for two possible $\boldsymbol{w}$:



Margin for suboptimal choice of $\boldsymbol{w}$



Maximum margin choice of $\boldsymbol{w}$

Bishop (2007), Fig. 7.1

The maximum-margin principle means we want to find

$$\underset{\boldsymbol{w},b}{\operatorname{argmax}} \left\{ \frac{1}{\|\boldsymbol{w}\|} \min_i [t_i(\boldsymbol{w}^T \phi(\boldsymbol{x}_n) + b)] \right\}$$

This can be simplified by rescaling the parameters to obtain the constrained optimization problem:

$$\underset{\boldsymbol{w},b}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{w}\|^2$$

subject to

$$t_i(\boldsymbol{w}^T \phi(\boldsymbol{x}_i) + b) \geq 1$$

for all $i = 1, \ldots, N$.

This is a standard quadratic programming problem.

In terms of the kernel function $k(\cdot, cdot)$ rather than the feature mapping function $\phi(\cdot)$, after optimization, we obtain the prediction function

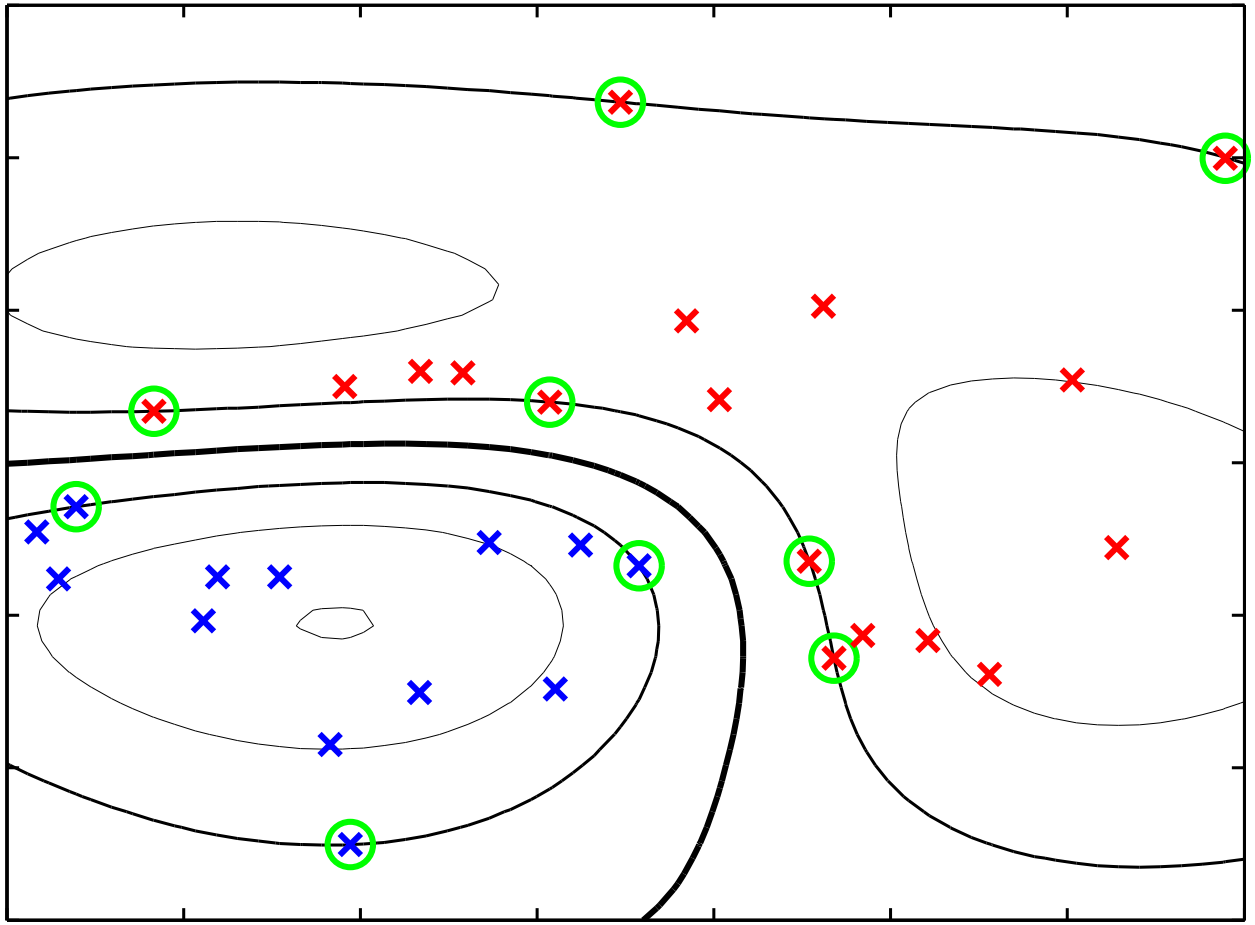
$$y(\boldsymbol{x}) = \sum_{i=1}^{N} a_i t_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

Most of the coefficients a_i will be 0, so they do not participate in classifying input data. The remaining non-zero multipliers correspond to training data points called the support vectors.

Denoting as $\mathcal{S}$ the indices of the support vectors (the set i such that $a_i > 0$), we can calculate the bias

$$b = \frac{1}{N_{\mathcal{S}}} \sum_{i \in \mathcal{S}} \left(t_i - \sum_{m \in \mathcal{S}} a_i t_i k(\boldsymbol{x}_n, \boldsymbol{x}_m) \right)$$

Example nonlinear decision boundary:



Bishop (2007), Fig. 7.2

Classification with SVMs

SVMs for the not linearly separable case

Depending on the distribution of $\{\boldsymbol{x}_i\}$ and the feature space mapping $\phi(\boldsymbol{x})$, the training data for the two classes might overlap in feature space.

In this case the constraints in the quadratic programming problem cannot be satisfied.

The standard method is to introduce slack variables ξ_i where $\xi_i = 0$ for training points on the correct side of the margin boundary and $\xi_i = |t_i - y(\boldsymbol{x}_i)|$ for incorrectly classified points.

This allows us to return to a constrained quadratic optimization problem.

Classification with SVMs

SVMs in practice

The typical approach is to use the ν-SVM with a radial basis function kernel of the form $k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|^2\right)$.

For multiclass problems, we typically construct $\frac{N(N-1)}{2}$ binary ν-SVMs then allow each classifier to "vote" for the class of a new input.

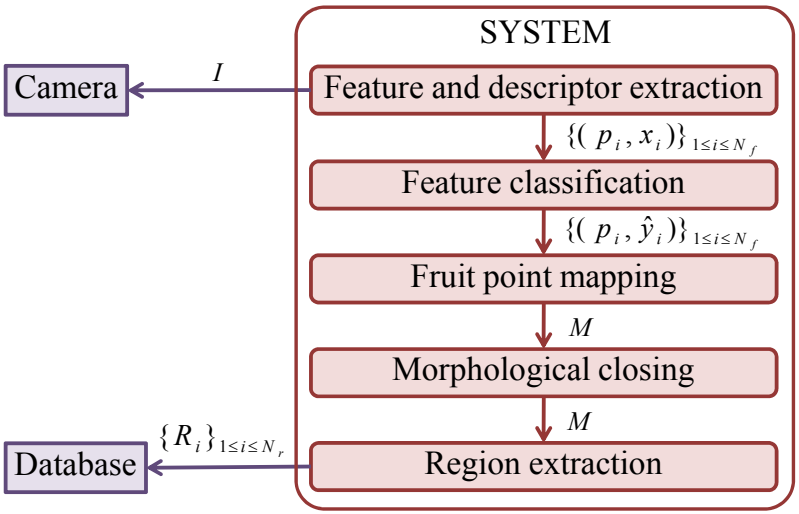
There is a very nice free implementation available called LIBSVM, with versions for Matlab, Perl, Ruby, C, Weka, and so on, available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

On Ubuntu just `apt-get install libsvm-tools`.

LIBSVM is also used in OpenCV.

Classification with SVMs

Example: fruit detection



Fruit detection system

Extracting feature points from agricultural image.

Classification with SVMs

Algorithm

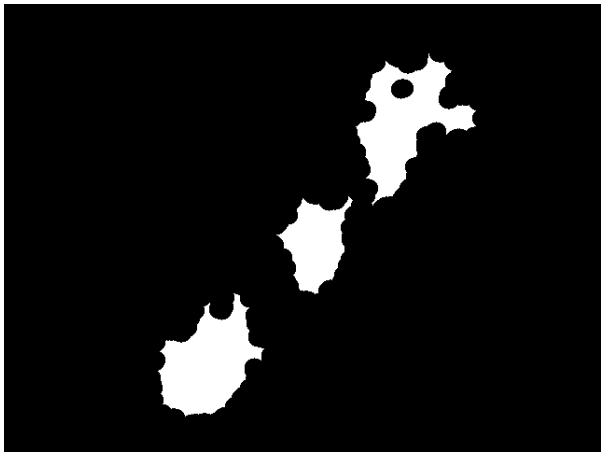
Algorithm:

1. Acquire input image I.
2. Apply interest point operator to obtain N_f candidate feature points $\{\boldsymbol{p}_i\}_{1 \leq i \leq N_f}$.
3. For each feature point $\boldsymbol{p}_i$, obtain a feature descriptor vector $\boldsymbol{x}_i$.
4. For each feature descriptor vector $\boldsymbol{x}_i$, obtain predicted label $\hat{y}_i$ (1 for positive or 0 for negative).
5. Create binary image M the same size as I and set all pixels to 0.
6. For each $\boldsymbol{p}_i$ for which $\hat{y}_i = 1$, set $M(\boldsymbol{p}_i)$ to 1.
7. Perform morphological closing with appropriately shaped structuring element on M.
8. Extract connected components from M and return large positive regions $\{R_i\}_{1 \leq i \leq N_r}$, as output.

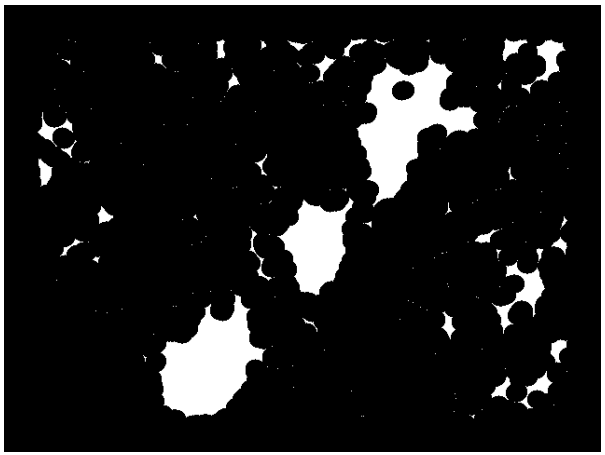
Classification with SVMs

Sample results

Predicted positive feature points



Connected components results



Filtering results

Detected regions

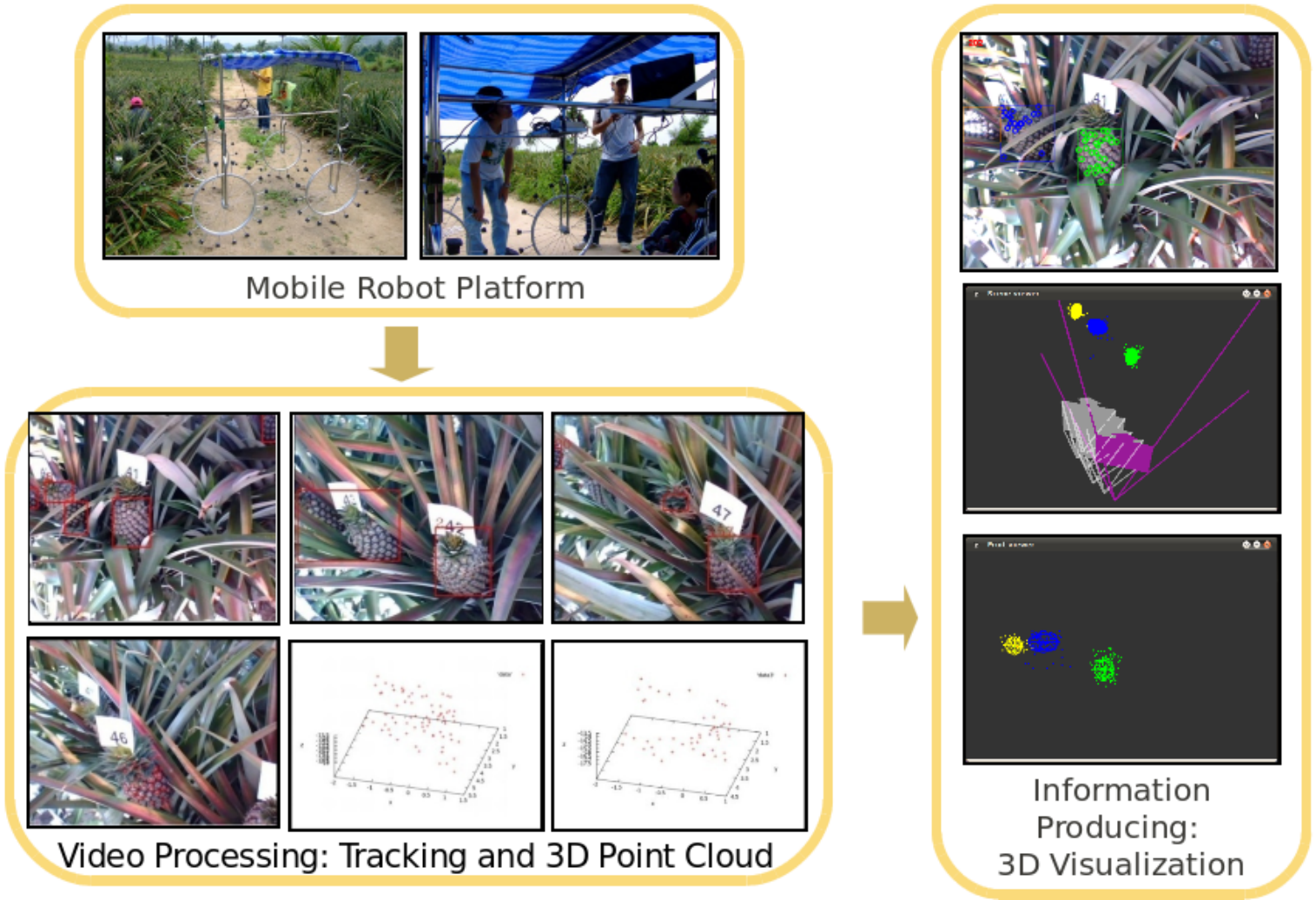
Outline

1. Preliminaries
2. Introduction to 3D computer vision
3. Texture modeling with SIFT
4. Classification with SVMs
5. Agricultural crop mapping
6. High-throughput plant phenotyping
7. Conclusion

Agricultural crop mapping

Mobile video processing for agricultural crop mapping

Current work in agricultural crop mapping:



Agricultural crop mapping

Mobile video processing for agricultural crop mapping

Agricultural crop mapping

Mobile video processing for agricultural crop mapping

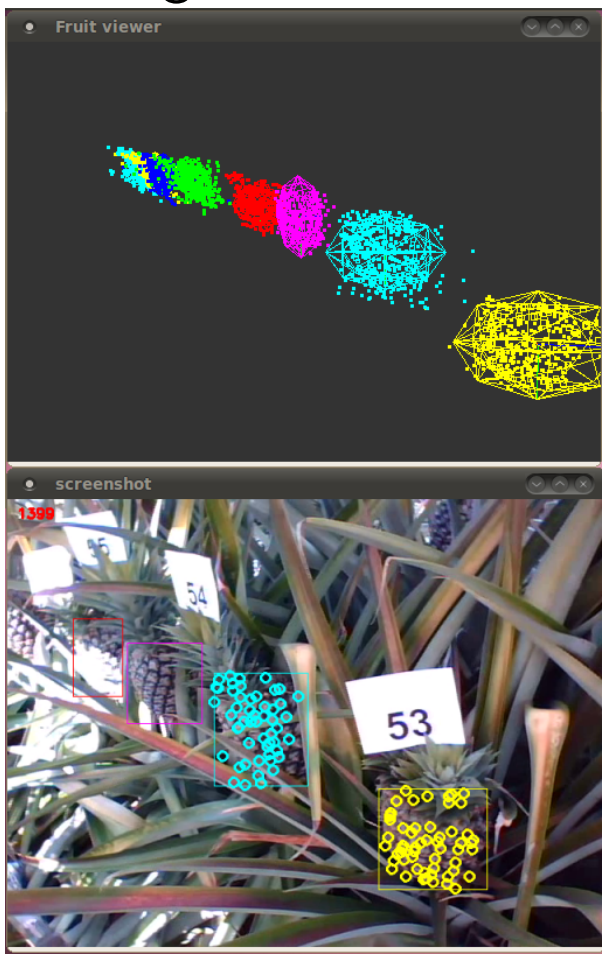
We aim to apply 3D vision and structure from motion for fruit in the field for purposes of yield prediction.

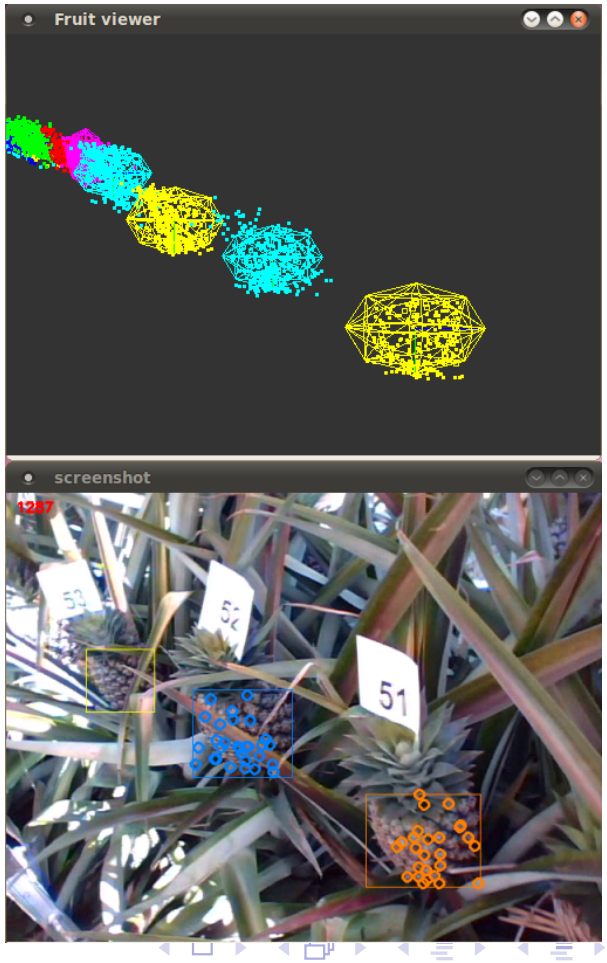
Image sequence extracted from video file taken from agricultural field

Agricultural crop mapping

Mobile video processing for agricultural crop mapping

We apply keyframe selection to the video sequence, find fruit regions (using keypoint descriptor classification), and perform 3D reconstruction of the point regions.





Agricultural crop mapping

Autonomous navigation for a quad-rotor UAV

This project uses a quad-copter with a low-resolution camera to gather videos of agricultural fields for inspection purposes.

Objectives:

- To obtain high-resolution image of a 3D scene using low quality, low-resolution monocular camera mounted on the quad-copter.
- To improve the quality of 3D-reconstruction by selecting key frames constructed using super-resolution.

Quad-copter

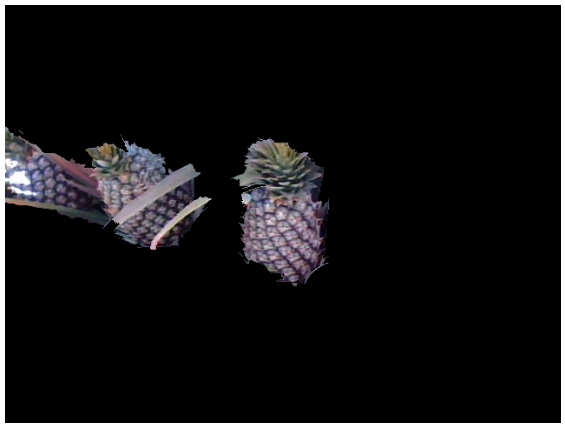
Image taken from quad-copter

Agricultural crop mapping

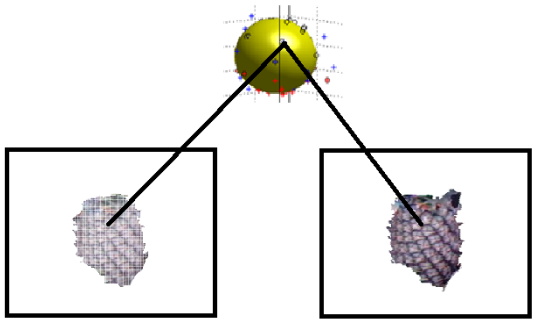
Autonomous navigation for a quadrotor UAV

We are developing simultaneous real-time navigation based on feature tracking and non-real-time 3D reconstruction.

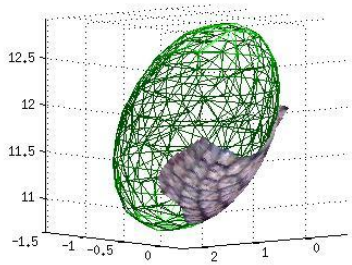
Eventually the work will enable yield prediction and detection of weed and pest infestation using the high-resolution frames constructed by super-resolution.



Dense segmentation with SIFT classifier



Optimization of spheroid to minimize SIFT distance

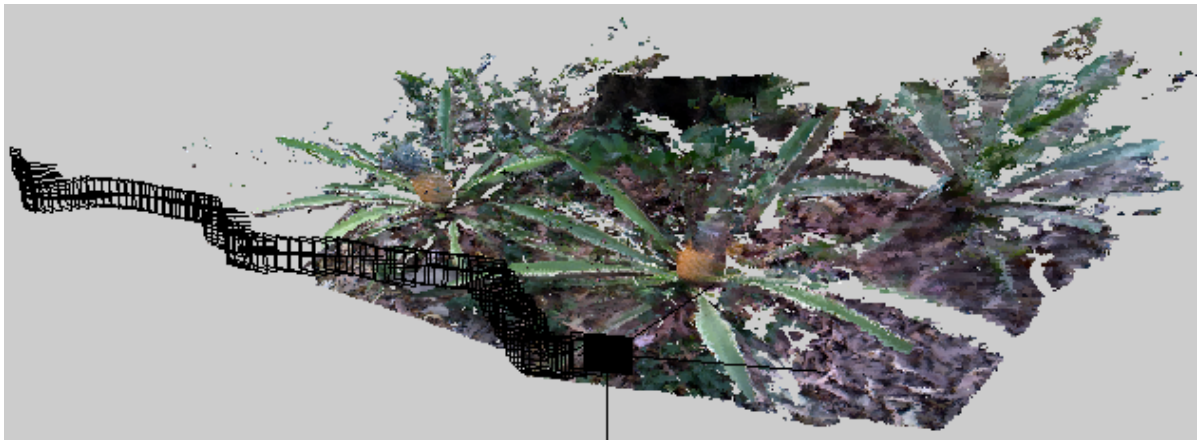


Optimized spheroid

Agricultural crop mapping

Fast fusion modeling of crops

We are working on Fast Fusion modeling of agricultural crops with RGBD sensors:



Outline

1. Preliminaries
2. Introduction to 3D computer vision
3. Texture modeling with SIFT
4. Classification with SVMs
5. Agricultural crop mapping
6. High-throughput plant phenotyping
7. Conclusion

High-throughput plant phenotyping

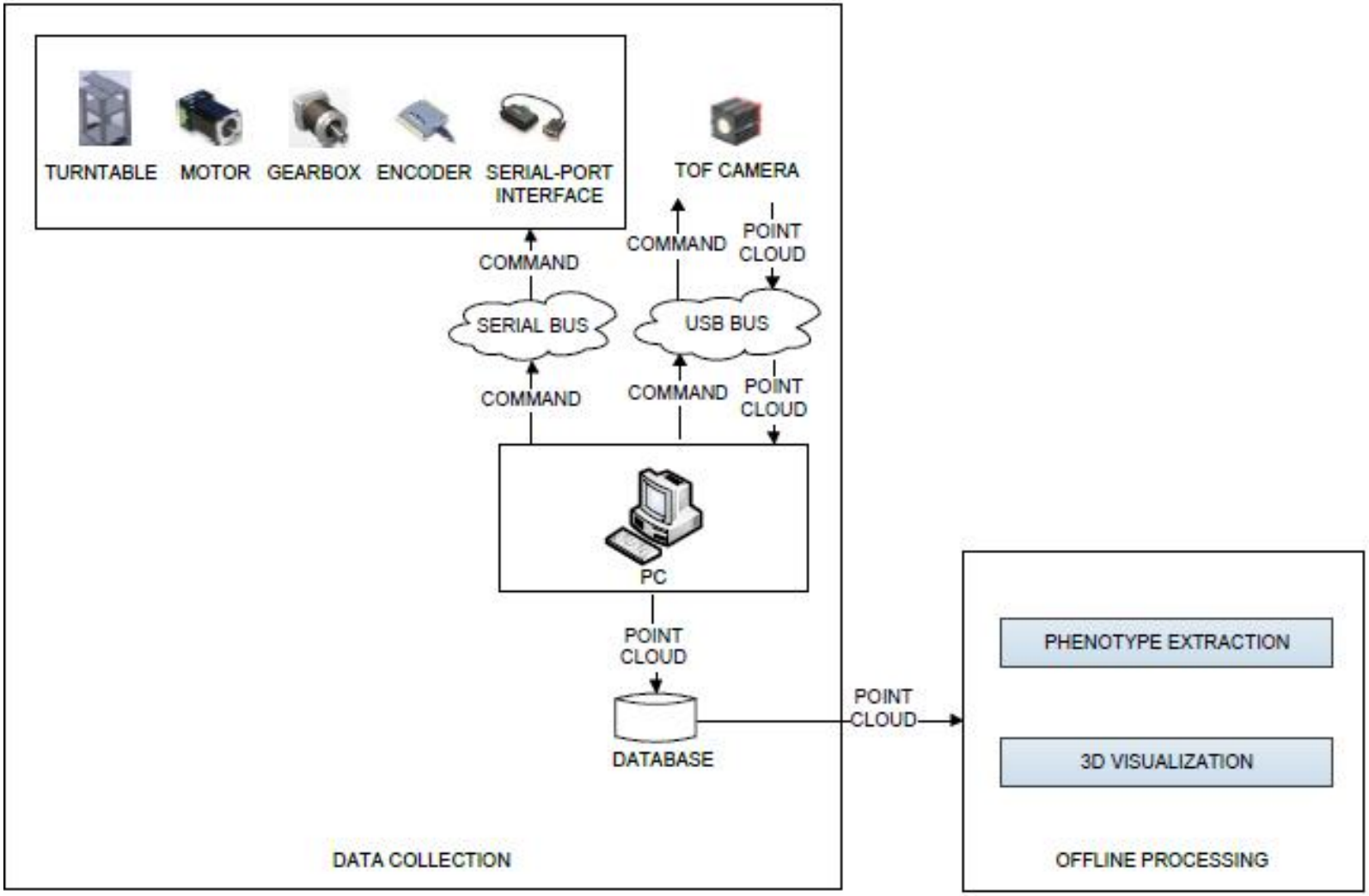
Introduction

Machine vision is also useful in the plant breeding process.

Breeders aim to obtain good seeds by selecting for desirable phenotypes.

Measuring plant phenotypes is tedious and error prone — video-based, non-destructive automation of the measurement process is desirable.

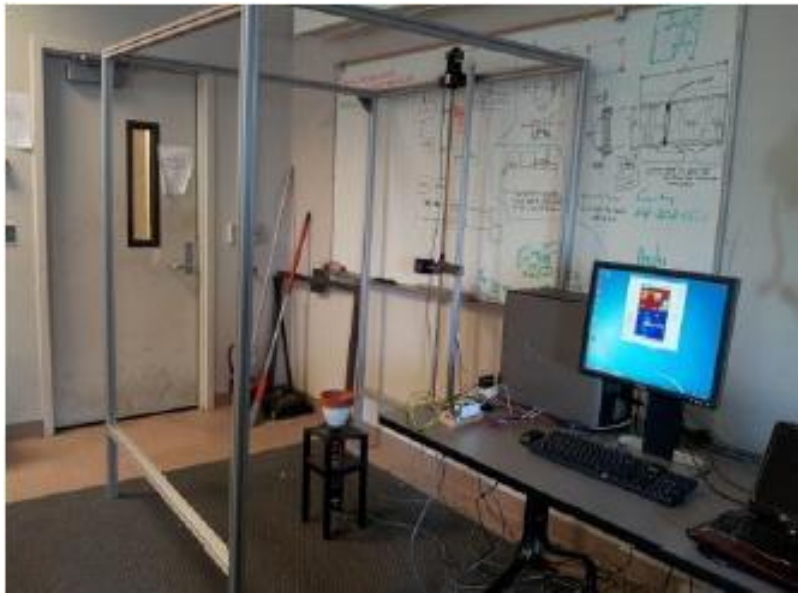
This is work with Supawadee Chaivivatrakul and colleagues at Iowa State University.



System architecture

High-throughput Plant Phenotyping

Prototype

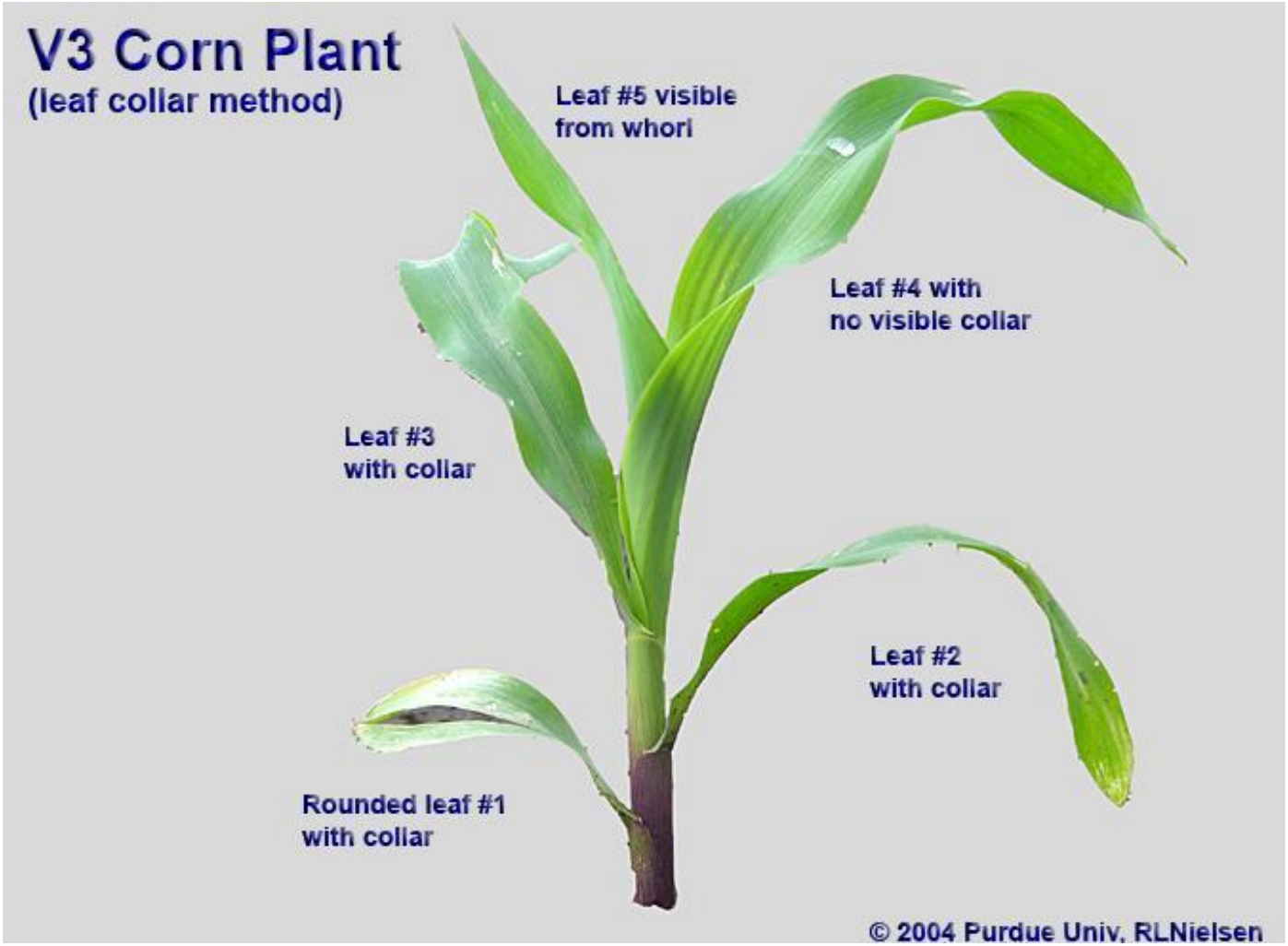


Prototype station

Young corn plant,
sampled with time-of-flight camera

High-throughput Plant Phenotyping

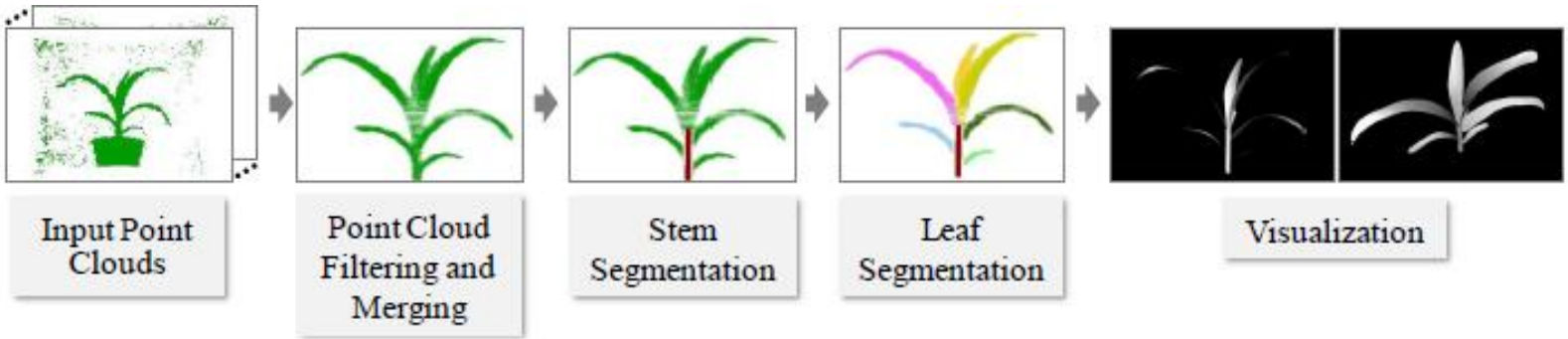
V3 corn plant



Reprinted from Nielsen (2014)

High-throughput Plant Phenotyping

Processing



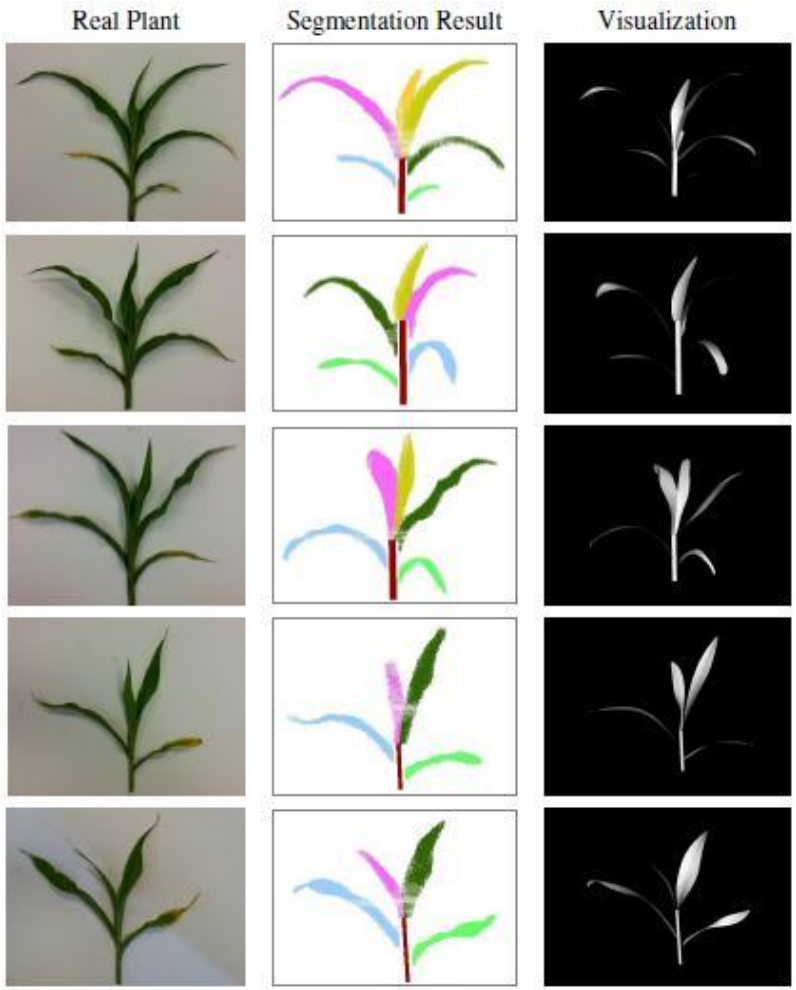
Processing pipeline

Overall error in phenotype measurement is not more than 22%:

Phenotype		Average error
Stem size	Major axis	7.92%
	Minor axis	15.2%
	Height	7.45%
Leaf area		21.89%
Leaf length		10.25%
Leaf angle	(plane fitting)	11.09%
	(skeletonization)	79.96%

High-throughput Plant Phenotyping

Results



Experimental results

[SHOW VIDEO]

Results with the prototype are promising enough for commercial explotation.

See Matt's Web page for more details (paper in *Computers and Electronics in Agriculture*).

Outline

1. Preliminaries
2. Introduction to 3D computer vision
3. Texture modeling with SIFT
4. Classification with SVMs
5. Agricultural crop mapping
6. High-throughput plant phenotyping
7. Conclusion

Conclusion

To conclude, most precision agriculture applications based on computer vision involve extraction of 3D or 2D structure and/or appearance.

Automatic extraction of structure is increasingly practical and well understood in the computer vision community.

Appearance extraction is more application specific.

In industrial environments, conditions can be controlled so that appearance is easy to measure.

In outdoor environments, simple problems such as illumination plague us.

Conclusion

Vision sensors are information-rich, cheap, and lightweight.

Mobile vision sensors require solutions to challenging AI problems.

A few of the open problems:

- Incremental localization and mapping algorithms are not yet as accurate as batch structure-from-motion algorithms in traditional computer vision.
- The higher level geometry of complex scenes is still not as accurate as we would like.
- The loop closing problem, which requires us to revise past state estimates, needs an elegant and efficient solution.

Conclusion

Current vision and robotics research at AIT

Current research in the Vision and Graphics Lab at AIT:

- Disease and pest classification for sugar cane crops (with Mitrpohl)
- Tree inventory monitoring using UAVs and SLAM/structure from motion (with EU-Tech)
- Many other machine vision applications!

Interested?

See `http://www.cs.ait.ac.th/~mdailey` for publications and more.

Scholarships are available for Master and Doctoral studies at AIT.