High demand for software engineers with computer vision experience

- Quality control and quality insurance become more important
  - Efficient and effective
  - Recall actions
- Availability of low cost vision solutions
  - PC + camera
  - Intelligent camera
  - Cellphone
- Using libraries (No development from scratch)
- Structured testing
- Evaluation of large datasets
- Applications
  - Industry
  - Gaming
  - Surveillance
  - Augmented reality
Demo

- Framework
- Find License Plate
- Find Characters
- Match Plate
- Lexicon

Overview

- Licenseplate recognition competition
  - Goal
  - Planning
  - Example
  - Rules
- Matching
  - Not using a lexicon
  - Using a lexicon
  - Rejecting classification results
- Framework
  - VisionLab
  - Components
  - Finding the license plate
  - Finding the characters
  - Reading the licenseplate
- C# Application
  - User interface
  - UML()
- Appendix
  - Generating the pattern matcher file
Goal

Next week (homework):
• Take 50 photo’s of license plates (each student)
• Taken with sensible white balance and exposure
• Different Angles / Lighting conditions
• All Dutch rectangular car license plates (yellow)
• One license plate per image (fully visible)

Before the end of the course (Deliverables)
1) C# software for reading license plates
2) Report, with focus on the creativity of your solution

Planning

Homework:
Week 1 – 4 : Theory and assignments
Week 5 : Finish LicensePlateMatcher.FindPlate() in C# or Finish find_plate.js in VisionLab
Week 6 : Finish LicensePlateMatcher.FindCharacters() in C# or Finish find_characters.js in VisionLab
Week 7 : Fully functional

Extra
- Questions
- Receive final set
- Receive minimum score

Midterm week
- Final competition
- Report
- No tweaking possible!
Examples

Rules

Not using the lexicon!

Scoring:

• 1 point for each correctly matched licenseplate
• 0 points for each unrecognized licenseplate
• 10 penalty points for each incorrectly recognized licenseplate

Grading:

1) Based on competencies of the course
2) Result of the competition on a selection of the photos
3) Minimum number of points will be determined after the selection of photos has been made
4) +1 for 1st and 2nd place (For the whole group)
Matching

No Lexicon

Approach
1. Match every character
2. Take character with the lowest error
3. Calculate confidence

Applications:
• Toll roads
• Speed camera’s
No Lexicon

<table>
<thead>
<tr>
<th></th>
<th>1st</th>
<th></th>
<th>2nd</th>
<th></th>
<th>3rd</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>O</td>
<td>0.07</td>
<td>3</td>
<td>0.10</td>
<td>M</td>
<td>0.21</td>
</tr>
<tr>
<td></td>
<td>O</td>
<td>0.18</td>
<td>5</td>
<td>0.22</td>
<td>H</td>
<td>0.22</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>0.20</td>
<td>S</td>
<td>0.24</td>
<td>N</td>
<td>0.28</td>
</tr>
</tbody>
</table>

Confidence:

|       | 0.61 | 0.61 | 0.18 | 0.59 | 0.75 | 0.44 |

03MZHZ is incorrect!

Lexicon

Approach
1. Match every character
2. Match license plate to all possible license plates
3. Take plate with the lowest error for a whole word
4. Calculate confidence

Applications:
• Limited entry
• Camp sites
• Car parks
Lexicon

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>3</th>
<th>H</th>
<th>Z</th>
<th>H</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>0.07</td>
<td>0.10</td>
<td>0.21</td>
<td>0.08</td>
<td>0.58</td>
<td>0.10</td>
</tr>
<tr>
<td>2nd</td>
<td>0.18</td>
<td>0.22</td>
<td>0.22</td>
<td>0.16</td>
<td>0.20</td>
<td>0.16</td>
</tr>
<tr>
<td>3rd</td>
<td>0.20</td>
<td>0.24</td>
<td>0.28</td>
<td>0.23</td>
<td>0.27</td>
<td>0.24</td>
</tr>
</tbody>
</table>

Confidence: 0.44

03MZHZ is not a word in the lexicon
03HZHZ is chosen instead, which is correct!

Rejection

1. Reject confidences below a certain value to detect mismatches.
2. Check ground-truth against the match result
3. Count instances for each category (TP, FP, FN, TN)
4. Score = True Positives - (10 * False Positives)

Confusion matrix

<table>
<thead>
<tr>
<th>Match result</th>
<th>Accept</th>
<th>Reject</th>
<th>Check with ground-truth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True</td>
<td>False</td>
<td>Correct</td>
</tr>
<tr>
<td></td>
<td>Positive</td>
<td>Negative</td>
<td></td>
</tr>
</tbody>
</table>

Confidence threshold is a tunable parameter:

- Decreasing will move False Negatives to True Positives which is good
- Decreasing will move True Negatives to False Positives which is bad
- The opposite is true for increasing
- When trying to improve the overall performance (True Positives and True Negatives) the vision method or parameters have to be improved
Framework

VisionLab

VisionLab is used for image processing (www.vdlmv.nl):

- Image processing algorithms
- Pattern matching
- Neural networks
- Genetic algorithms
- Algorithms written in ANSI C++
- OpenMP
- OpenCL
- Portable software:
  - Windows, Linux and Android
  - x86, x64, ARM and PowerPC
Components

- **LicensePlate.cs**
  - Public class `LicenseCharacter`
    - Public `LicenseCharacter(string character, double error, double confidence)`
    - Public string `character()`
    - Public double `error()`
    - Public double `confidence()`
    - Public new string `ToString()`

- **VisionLabEx.cs**
  - C# C++ compatibility functions for VisionLab

- **VisionLab class**
  - Computer Vision library

- **LicensePlateMatcher.cs**
  - License plate datastructures

- **LicensePlateMatcherScript.cs**
  - Functions for the actual license plate recognition

- **frmMain.cs**
  - GUI, Initialization, main loop and administration

**LicensePlate.cs**

```csharp
public class LicenseCharacter {
    public LicenseCharacter(string character, double error, double confidence)
    public string character()
    public double error()
    public double confidence()
    public new string ToString()
}

public class LicensePlate {
    public LicensePlate()
    public double confidence
    public List<LicenseCharacter> characters
    public string getLicensePlateErrorsString()
    public string getLicensePlateString()
    public new string ToString()
}
```

26-8-2014 License plate recognition 17
LicensePlateMatcher.cs
Functions for finding and reading license plate using VisionLab C# library

```csharp
public class LicensePlateMatcher {
    public static bool FindPlate(RGB888Image platelimage, ref Int32Image binaryPlatelimage);
    public static bool FindCharacters(RGB888Image platelimage, Int32Image binaryPlatelimage, ref Int32Image labeledRectifiedPlatelimage);
    public static bool MatchPlate(Int32Image binaryRectifiedPlatelimage, BlobMatcher_Int32 matcher, ClassLexicon lexicon, ref LicensePlate result, ref LicensePlate lexiconResult);
}
```

LicensePlateMatcherScript.cs
Functions for finding and reading license plate using VisionLab .JL scripts

```csharp
public class LicensePlateMatcher {
    public static bool Init() // Read scripts and pattern matcher
        // initialize command interpreter
    public static bool FindPlate(RGB888Image platelimage, ref Int32Image binaryPlatelimage);
    public static bool FindCharacters(RGB888Image platelimage, Int32Image binaryPlatelimage, ref Int32Image labeledRectifiedPlatelimage);
    public static bool MatchPlate(Int32Image binaryRectifiedPlatelimage, ref LicensePlate result);
}
```
Production phase license recognition

Finding the license plate
public static bool FindPlate ()

Description:
Find the largest license plate in the image
1. Segment using ThresholdHSVchannels
2. Remove blobs which are not license plates

Input:
// Original image
RGB888Image platImage

Output:
// Segmented license plate
ref Int32Image binaryPlateImage

Return:
// License plate found?
bool

IsServerScript

SERVERSCRIPT = IsServerScript
if SERVERSCRIPT == false
    // Copy script selected image (F6)
    copy %currentimage OriginalImage
    else
    // Copy first image passed by C#
    copy %p1 OriginalImage
endif

if SERVERSCRIPT == false
    display LicensePlateBin
    else
    // Copy result image back to C#
    copy LicensePlateBin %p2
endif

If IsServerScript returns true, the script is being executed from C#.
If IsServerScript returns false, the script is being executed from the VisionLab Client.

This can be used to autodetect if images are passed by C# or if selected images in VisionLab should be used. This is convenient for debugging scripts.
LicensePlateMatcher.FindPlate()

```csharp
public static bool FindPlate( RGB888Image plateImage, ref Int32Image binaryPlateImage ){
    const int c_threshold_h_min = 21;
    const int c_threshold_h_max = 50;
    const int c_threshold_s_min = 100;
    const int c_threshold_s_max = 255;
    const int c_threshold_v_min = 100;
    const int c_threshold_v_max = 255;
    const int c_remove_blobs_min = 1;
    const int c_remove_blobs_max = 5000;

    HSV888Image plateImageHSV = new HSV888Image();
    //Convert to RGB to HSV
    VisionLab.Convert(plateImage, plateImageHSV);
    //*******************************//
    //** Exercise:                 **//
    //**   adjust licenseplate    **//
    //**   segmentation           **//
    //*******************************//
    //Threshold HSV image
    VisionLab.Threshold3Channels( plateImageHSV, binaryPlateImage , c_threshold_h_min, c_threshold_h_max,
    c_threshold_s_min, c_threshold_s_max,
    c_threshold_v_min, c_threshold_v_max);
    //Remove blobs with small areas
    VisionLab.RemoveBlobs( binaryPlateImage, Connected.EightConnected,
    c_remove_blobs_min, c_remove_blobs_max);
    //Fill up characters
    VisionLab.FillHoles(binaryPlateImage, Connected.FourConnected);
    plateImageHSV.Dispose();
    //Return true, if pixels found
    return (VisionLab.SumIntPixels(binaryPlateImage) > 0);
}
```

License plate recognition
public static bool FindPlate( RGB888Image plateImage, ref Int32Image binaryPlateImage ){
    //Upload image to VisionLab command interpreter
    VisionLab.SetRGB888Image(cmdInt, "plateImage", plateImage);
    //Execute script using the "icall" command
    String result = StripTime(
        cmdInt.ExecRequest("icall FindPlate plateImage, binaryPlateImage")
    );
    //Download result image from VisionLab command interpreter
    VisionLab.GetInt32Image(cmdInt, "binaryPlateImage", binaryPlateImage);
    //Return if the script returned "true" or "false"
    return (result == "true");
}

Solutions brainstorm for finding the license plate

1. Tune ThresholdHSVChannels
   Find darkest and brightest yellow license plates
   Analyse HSV values, apply values, test values

2. Tune RemoveBlobs
   Find smallest and largest licenseplate
   Analyse Area, apply criteria, test criteria

3. Add additional criteria
   Add RemoveBlobs line using LengthBreadthRatio as a feature
   Add RemoveBlobs using additional features

4. Add additional segmentation functions
   Use a different alternative Threshold values

5. Etcetera
   This will score points
Finding the characters

public static bool FindCharacters ()

Description:
Locates the characters of the license plate
- Warp image (Rectify)
- Segment characters
- Remove blobs which are to small (Lines between characters)

Input:
//Original image
RGB888Image platelImage
//Segmented license plate
Int32Image binaryPlateImage

Output:
//Image containing binary six characters
ref Int32Image binaryCharacterImage

Return:
//Function executed successfully
bool
Solutions brainstorm for finding characters

1. Try FindCornersRectangleSq operator/function instead of FindCornersRectangle

2. Tune RemoveBlobs
   - Find smallest and largest license plate
   - Analyse Area, apply criteria, test criteria

3. Add additional criteria
   - Add RemoveBlobs using additional features

4. Add additional segmentation functions
   - Use a few different Threshold values and methods (manual vs. automatic)

5. Use a background correction
   - Subtract the background

6. Use a different color space
   - Use HSV

7. Use binary morphological filters to dilate or erode blobs
   - Try making the blobs more like the characters in the .pm file

8. Etcetera
   - This will score points

26-8-2014 License plate recognition 32
Reading the license plate

26-8-2014 License plate recognition

public static bool MatchPlate ()

Description:
Read the license plate

Input:
//Rectified license plate image containing six characters
Int32Image labeledRectifiedPlateImage
BlobMatcher_Int32 matcher    //initialized blobmatcher
ClassLexicon lexicon        //initialized lexicon

Output:
//Result by the blob matcher
ref LicensePlate result
//Result by the lexicon
ref LicensePlate lexiconResult

Return:
//six characters found?
bool

03 Hz Hz
match_plate.jls
(Part I)

```plaintext
$ServerScript = IsServerScript
if $ServerScript === false
  //Copy third selected image (Operator->Select 3rd)
copy %thirdimage LicensePlateRectifiedBin
$lpwd = lpwd
cwd $lpwd
PM_ReadFromFile PatternMatcher ../VL/lic_fonts.pm
else
  copy %p1 LicensePlateRectifiedBin
Endif

//Analyze blobs locations
copy LicensePlateRectifiedBin LicensePlateRectifiedLabel
labelblobs LicensePlateRectifiedLabel EightConnected
$maxBlobIndex = BlobAnalysisArray
$tab SortDown TopLeft UseX Height TopLeft Width
if $maxBlobIndex !== 5 then //Check if 5 characters were found
  return false
Endif

...  

match_plate.jls
(Part II)

```
Solutions brainstorm for reading the license plate

1. Adapt pattern matcher (.pm file)
   Analyze which characters occur in real license plates
   Use a different font in the .pm file

2. Check or correct license plate grammar
   D3-HF-BR is not very likely to be a license plate, while 03-HF-BR is

3. Use separate matchers for numbers and letters
   Use two .pm files

4. Etcetera
   This will score points

Framework
1. Images to be processed
2. Current score
3. Begin processing
4. Output images from LicensePlateMatcher (Script)
5. Reject below this confidence divided by 10
6. Match result from current licenseplate
7. Functions return values for these licenseplates
8. True Positive (Correct match and high confidence)
9. Reject (Confidence is too low)
10. False Positive (Confidence is high, but match is wrong)
public void frmMain.DisplayBlobs ()

Description:
Display features of the blob to the debugging output of C#

Input:
//Binary image containing the blobs
Image binaryImage

Effect:
//Blob features to the output
* Note: only works in debug mode
public void DisplayBlobs(Image binaryImage)
{
    vector_BlobAnalysis ba = new vector_BlobAnalysis();
    vector_Blob blobs = new vector_Blob();
    ba.Add(blobAnalysis.BA_Area);
    ba.Add(blobAnalysis.BA_Eccentricity);
    ba.Add(blobAnalysis.BA_LengthBreadthRatio);
    VisionLabEx.GetBlobsInfo(binaryImage, ba, ref blobs);
    System.Diagnostics.Debug.WriteLine("Blob Info: ");
    foreach (Blob b in blobs)
    {
            " Eccentricity: " +
            b.Eccentricity.ToString() +
            " LengthBreadthRatio: " +
            b.LengthBreadthRatio.ToString());
    }
    ba.Dispose();
    blobs.Dispose();
}
PM_CreateBlobMatcher pm Int16Image 60 1 20 0
$names = 0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
VarToArray $names $nameTab
lread allPats lic_fonts.jl
copy allPats allPatsB
ThresholdIsodata allPatsB DarkObject
$nNums = LabelBlobs allPatsB EightConnected
$nMaxBlob = BlobAnalysisArray allPatsB $tab SortDown TopLeft UseX Height TopLeft Width
for $i = 0 to $nMaxBlob do
VarToArray $tab[$i] $elm
$label = $elm[0]
$h = $elm[1]
$tl = $elm[2]
$w = $elm[3]
$x = getnthfromvector 1 $tl
$y = getnthfromvector 2 $tl
ROI allPats roi $x $y $h $w
copy roi roiB
Threshold roiB 0 100
PM_AddPattern roiB pm $nameTab[$i]
endfor
PM_WriteToFile pm lic_fonts.pm
PM_Delete pm

26-8-2014 License plate recognition 45