idneo

You dream the future,
we design your dream

Development of an artificial vision system to automatically inspect blood typing cards

Marc Blanch
R&D Hardware supervisor
Hardware Medical

Martí Cobos
R&D Hardware engineer
Hardware Medical

30th May 2018


Our mission:
We Design your Dreams

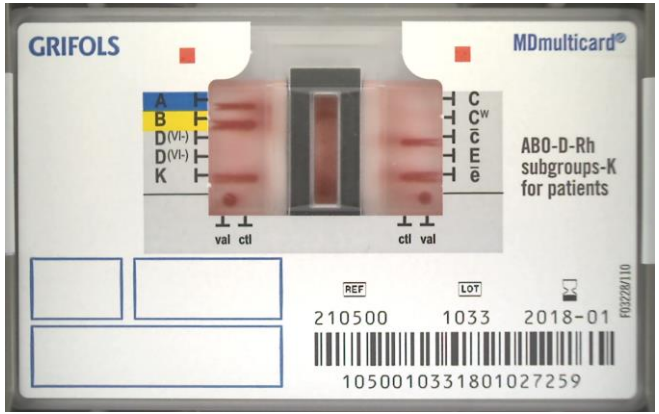
Outline

1. Key takeaways
2. Introduction to Organization and Business
3. Innovation Challenges and Achievements
4. How did we get there and leverage MathWorks
5. Further details on solution adopted
6. Concluding Remarks

idneo

Key takeaways

1. Automated diagnostics for blood typing cards made by a machine vision algorithm involve complex band analysis. Several intrinsic and extrinsic factors involved
2. Need to evaluate system performance using mathematical/statistical tools.
3. Portability of the machine vision system using MATLAB Coder and Embedded Coder to an embedded system powered by an ARM A9 processor running Android OS.



idneo

Who We are

Founded 2010

Barcelona HQ

+350 Engineers

ISO-9001 / ISO-13485
ISO-TS16949 / ISO-IEC 17025

Europe – USA -Asia

Idneo is an engineering services company focused in complete product development and working with customers form various sectors of innovative technologies, including Medtech, automotive, industrial, and consumer electronics.

We are a company strongly focused and dedicated to support development of innovative technological advancements.

IDNEO's potential is that we integrate all product development technical disciplines and expertise in-house which facilitates the process of going from an idea to a product.

idneo

Our services

All value chain at same partner
Speed to industrialization Quality-focused

Mechanical Design
Electronics Design
Software Design
Vision Systems Design

CE Mark
FCC
CB Scheme
EMC consulting

CONCEPT DEFINITION

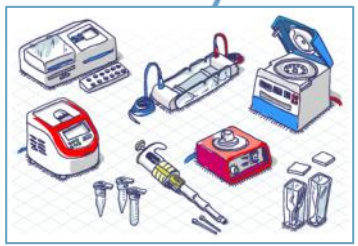
PRODUCT DEVELOPMENT

VALIDATION

PRODUCT COMPLIANCE

MANUFACTURING

Industrial design
Prototyping
Ergonomics
Usability



EMC
Product Safety
Reliability
Functional Testing

Industrialization
Short & large series

idneo

IDNEO Vision in computer vision systems development

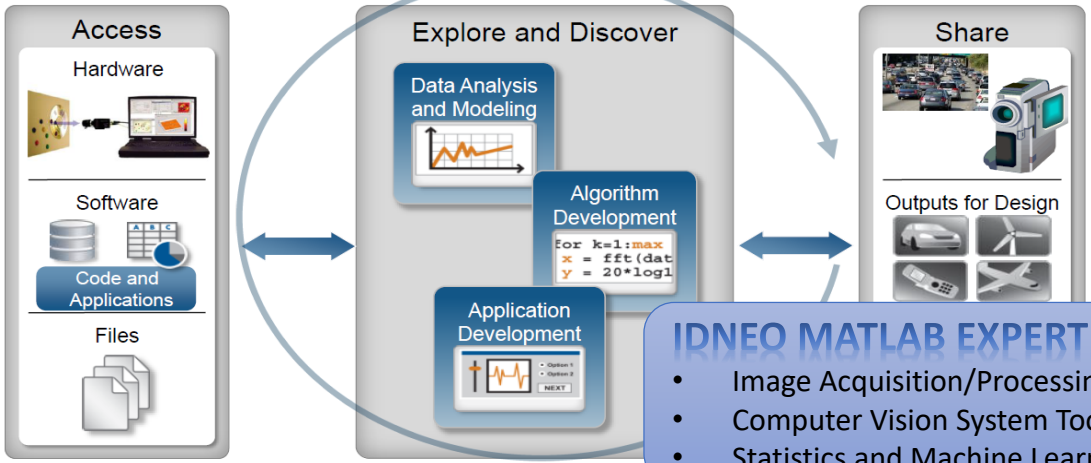
COMPETITORS

- Bound to proprietary hardware solution
  - Smart vision sensors
  - Smart vision systems
- Bound to limited proprietary software solution in specific platform.

IDNEO

- Customized solution, non-bound to target platform (any HW, any OS)
- Focus on application & image processing algorithms

MATLAB TOOLBOXES &
MATLAB CODER + EMBEDDED CODER

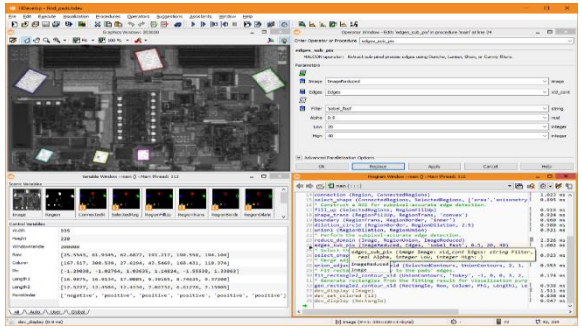


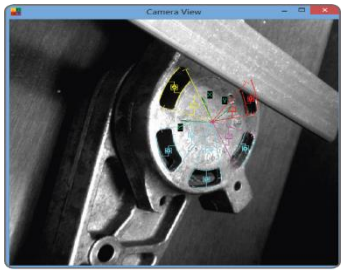
IDNEO MATLAB EXPERTISE

- Image Acquisition/Processing Toolbox
- Computer Vision System Toolbox
- Statistics and Machine Learning Toolbox

Embedded Focus: MATLAB vs Standard Computer Vision Libraries

OPEN CV (FOSS)
COGNEX VISIONPRO
HALCON
MATROX IMAGING LIB (MIL)
NI VISION
SCORPION VISION





CV LIBRARIES

Advantages:

- Cost-of-ownership
- Native code generation

Disadvantages:

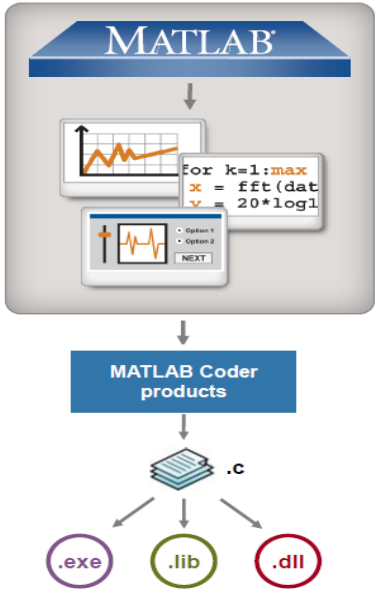
- High complexity, lots of different suites to learn
- Limited algorithms
- Less user-friendly visualization and debugging tools
- Weak documentation and little community support

MATLAB

Advantages:

- Unlimited function set
- Unlimited algorithms combined with other MATLAB toolboxes (machine learning, signal processing, etc)
- Visualization and user-friendly debugging tools
- Great documentation

MATLAB + Embedded Coder



idneo

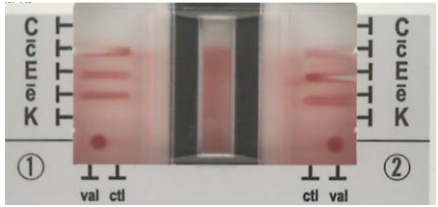
Innovation Challenges and Achievements

Challenges

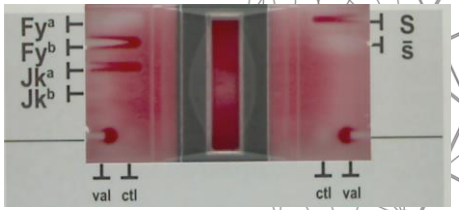
- Develop an algorithm for blood typing cards automatic analysis. Band classification algorithm to be developed with little knowledge of bands properties → Only a few card images with bands interpretation classified by an expert were initially provided.
  - Boundaries between positive, negative, doubtful and mixfields (mixed original-transfused blood) interpretations very challenging.
  - Band color intensity differed between card types.
  - Band localization needs to be addressed for every card using card fiducials.
  - High variability coming from manual blood dilution process.
  - Strong influence on extrinsic parameters like temperature, humidity, blood properties variation over time after reaction.
  - Strong dependence on biologist interpretations, even in doubtful cases expectations system works in all conditions.

Reaction	Pattern	Sample
Negative	No band	
	White band on red background	
Doubtful	Weak result	
	Shadow band	
	Partial red perimeter of band	
Positive	Solid red band	
	Well defined red perimeter of band	

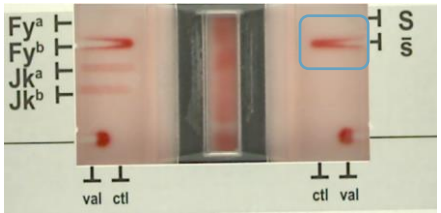
BAND PATTERNS



BAD DISPENSING



DIFFERENT INTENSITY BKG



MIXED BLOOD

FIDUCIALS LOCALIZATION

Innovation Challenges and Achievements

Challenges

- Integrate machine vision system into an embedded system with an android OS.
- Necessity to show evidences and proof of system results
- Complete system development, not only machine vision software algorithms, but also acquisition system selection (camera and illumination system).
- Tight project schedule, small team and limited resources to develop the project.

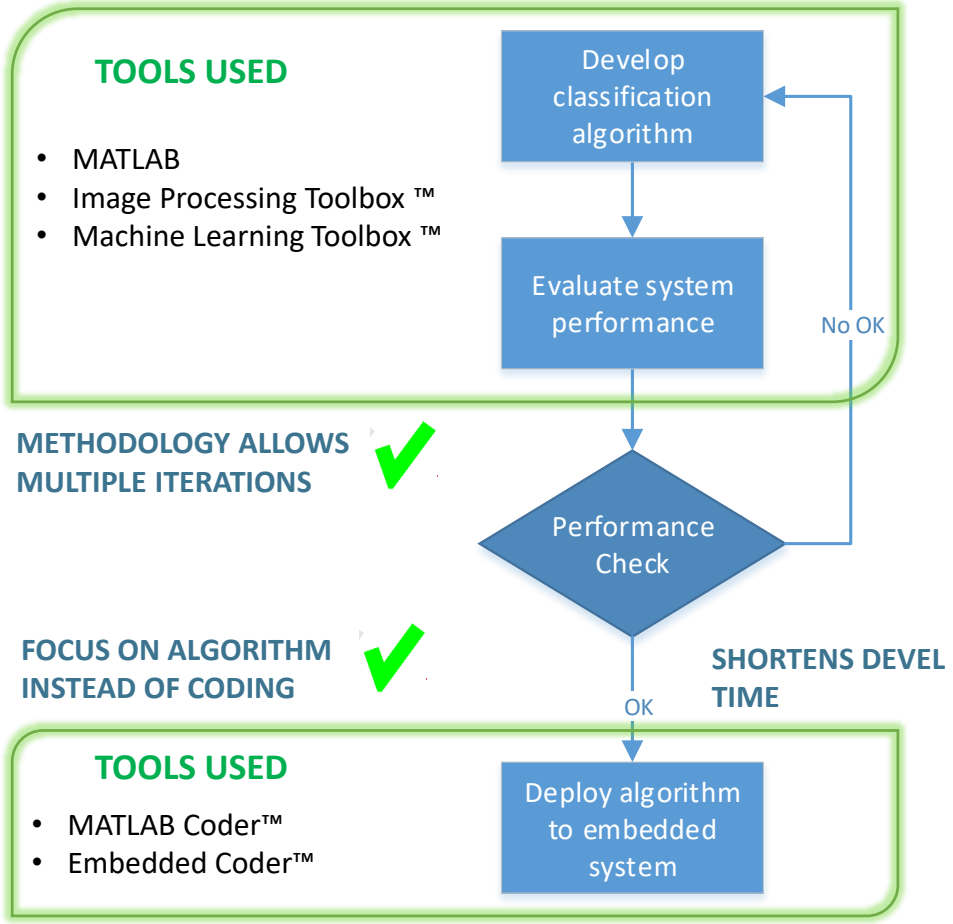
Achievements

- The system was developed iteratively following an agile methodology within customer strict schedule expectations.
- Challenging band classification system was developed with the ability to classify bands within 4 different types (negative, positive, doubtful, mix-field)
- Classification system was implemented using machine learning techniques but with a low number of features extracted from bands allowing traceability for algorithm decision tree in every diagnostics proposal.
- Artificial vision system performance exceeded customer expectations.
- System performance evaluation software was developed for customer to asses performance without development tools & metrics were proposed to quantify performance of different algorithm versions.

idneo

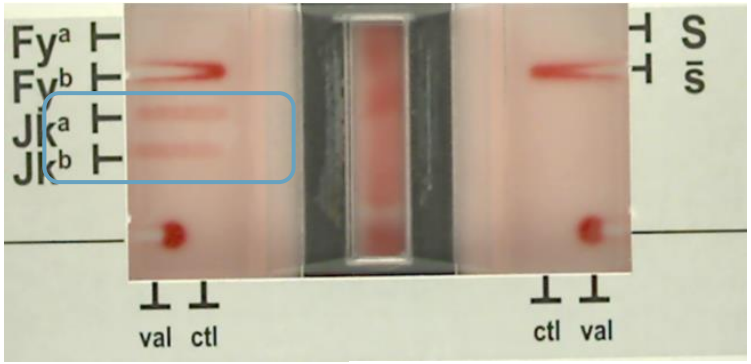
Development process

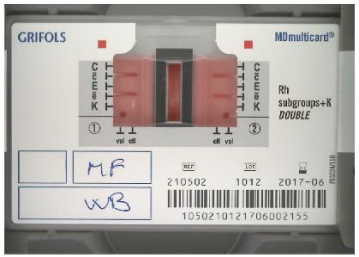
- Initially, the image processing algorithm is developed on MATLAB using Image Processing Toolbox.
- System performance is assessed with a dataset of more than 1200 images of different types of cards
- If the system doesn't reach the performance requirements the algorithm is refined, iteratively
- Once the system performance achieves customer requirements, it is deployed to the embedded system by converting the code to C language.

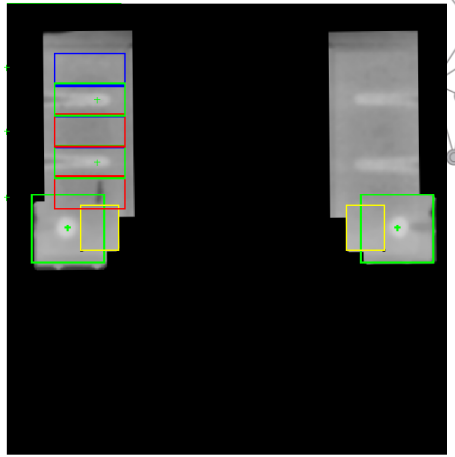


Development process – Image processing algorithm development

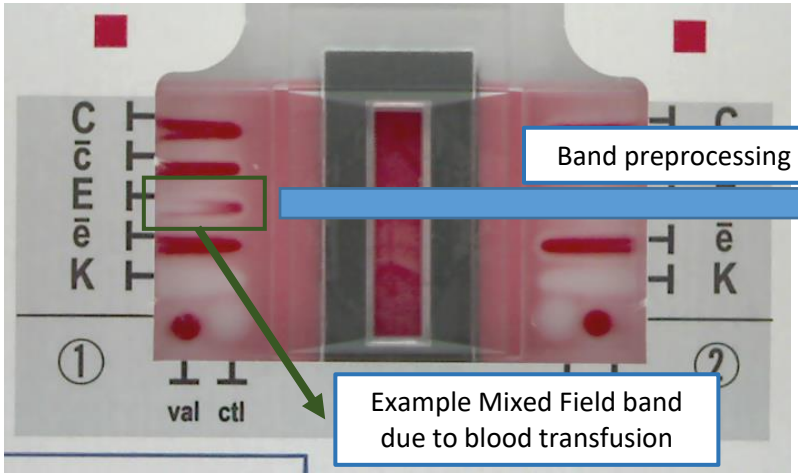
1. Perform white balance and color equalization
2. Convert image to Luv color space
3. Compute color difference to pure white
4. Search for card fiducials
5. Detect position and analyze ctl and val dots
6. Detect and analyze bands







Development process – Band Analysis – Mixed Field bands



Band preprocessing

Example Mixed Field band due to blood transfusion

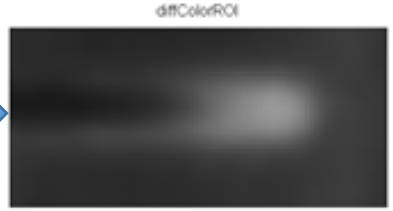
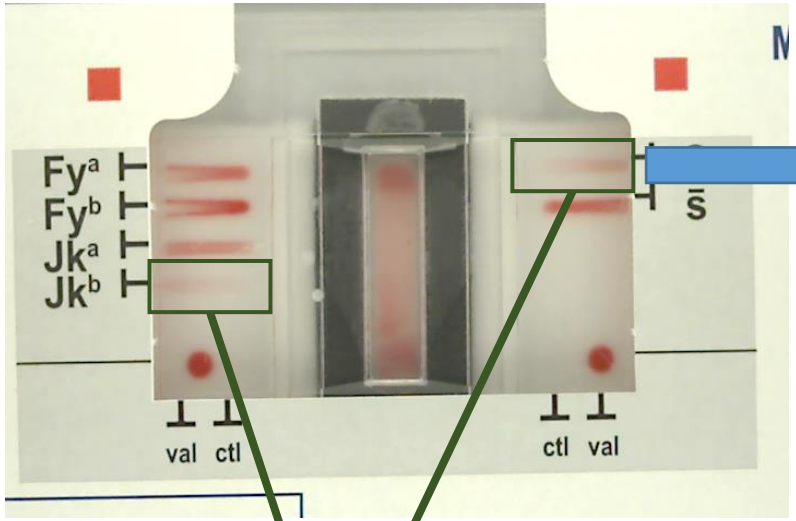


Image binarization using Otsu method (graythresh)

Perform skeleton binary morphological operation

Analyze band skeleton to classify image between positive or mixed field

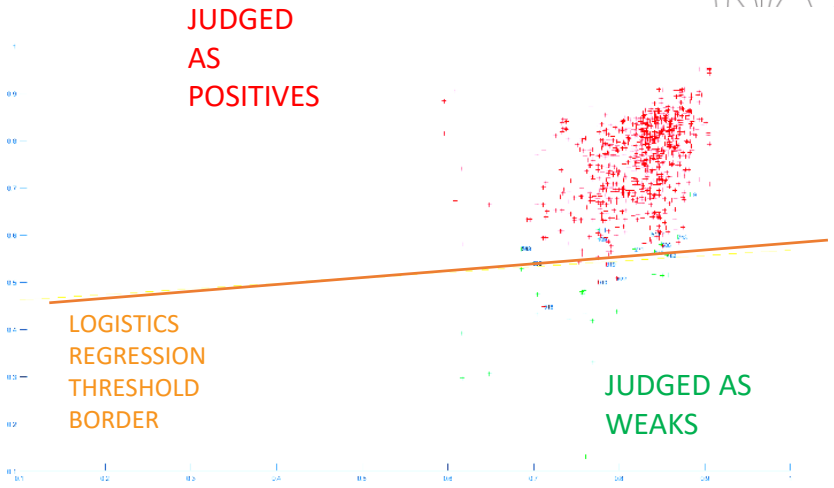
Development process – Band Analysis –Weak Bands



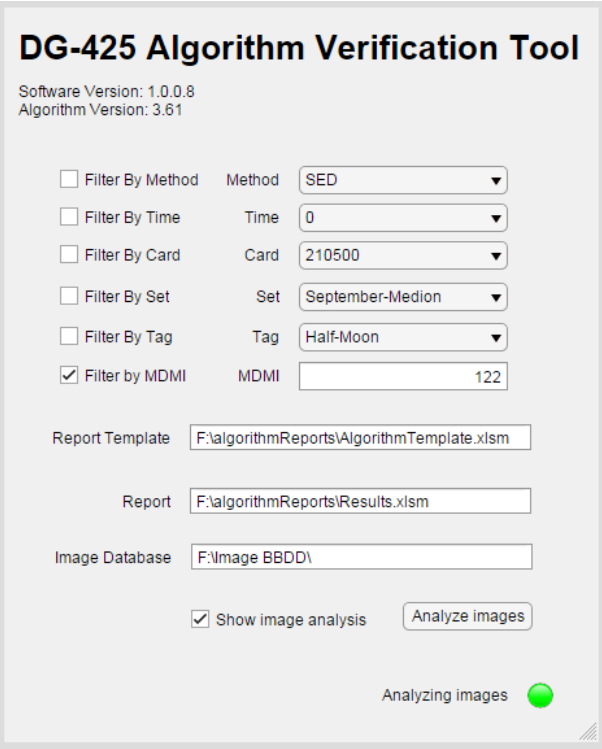
Weak bands

Color intensity feature extraction

Logistic regression classification



Development process - Automated verification tool



With the use of MATLAB App Designer and MATLAB Compiler a verification application was developed to asses system performance without the need of MATLAB.

Main functionalities:

- Analyze cards and store results
- Evaluate system performance (Accuracy)
- Compute confusion matrix

Effortless image processing algorithm integration into the verification application.

Compute Confusion matrix

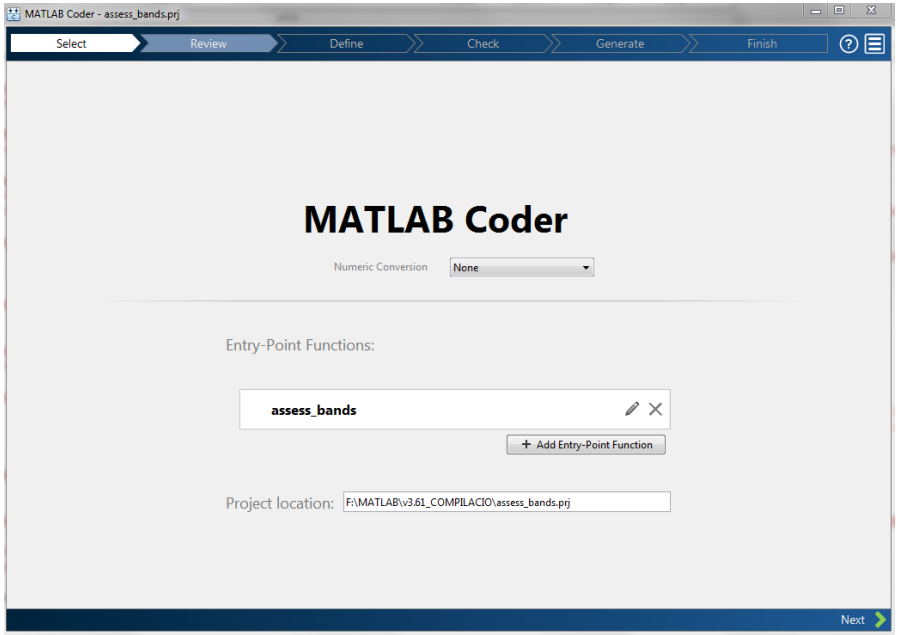
Band Analysis

Algorithm results	Expected results: Doubtful	Mixedfield	+	-
Doubtful	442	600	45	86
Mixedfield	8	93	9	0
+	55	42	3941	1
-	13	6	1	3783

Development process – Deploying the algorithm into an embedded system

Image processing algorithm development

.m



C code

.c

- Integration test and unit test run in a continuous integration system with *.c files
- From .m files to C code easy to integrate into an embedded processor within minutes.

idneo

Further details on solution adopted

Development process – Classification Learner App (Statistics and Machine Learning Toolbox)

Custom machine learning development

Logistic regression:

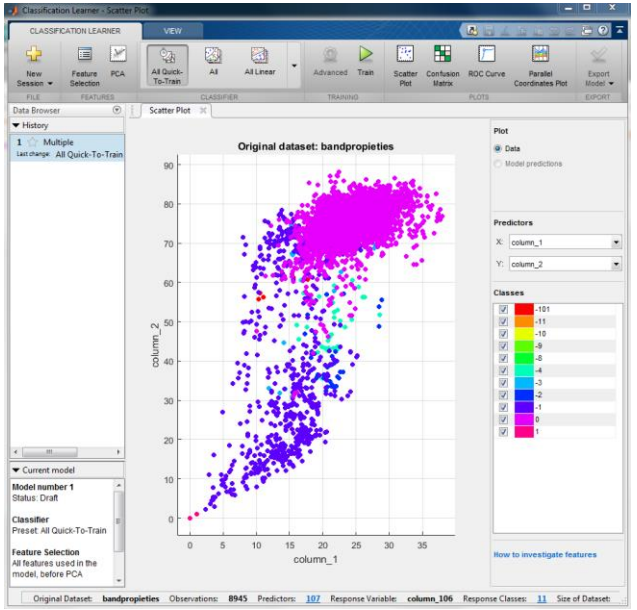
```
i = 0;                                            % zero the run length counter
ls_failed = 0;                                    % no previous line search has failed
fX = [];
[f1 df1] = eval(argstr);                          % get function value and gradient
i = i + (length<0);                                                % count epochs?!
s = -df1;                                         % search direction is steepest
d1 = -s'*s;                                                % this is the slope
z1 = red/(1-d1);                                  % initial step is red/(|s|+1)

while i < abs(length)                                              % while not finished
  i = i + (length>0);                                              % count iterations?!

  X0 = X; f0 = f1; df0 = df1;                     % make a copy of current values
  X = X + z1*s;                                                % begin line search
  [f2 df2] = eval(argstr);
  i = i + (length<0);                                              % count epochs?!
  d2 = df2'*s;
  f3 = f1; d3 = d1; z3 = -z1;             % initialize point 3 equal to point 1
  if length>0, M = MAX; else M = min(MAX, -length-i); end
  success = 0; limit = -1;                         % initialize quanteties
  while 1
    while ((f2 > f1+z1*RHO*d1) | (d2 > -SIG*d1)) & (M > 0)
      limit = z1;                                                  % tighten the bracket
      if f2 > f1
        z2 = z3 - (0.5*d3*z3*z3)/(d3*z3+f2-f3);                    % quadratic fit
      else
        A = 6*(f2-f3)/z3+3*(d2+d3);                                % cubic fit
        B = 3*(f3-f2)-z3*(d3+2*d2);
        z2 = (sqrt(B*B-A*d2*z3*z3)-B)/A;       % numerical error possible - ok!
      end
      if isnan(z2) | isinf(z2)
        z2 = z3/2;                  % if we had a numerical problem then bisect
      end
      z2 = max(min(z2, INT*z3),(1-INT)*z3);  % don't accept too close to limits
      z1 = z1 + z2;                                                % update the step
      X = X + z2*s;
      [f2 df2] = eval(argstr);
      M = M - 1; i = i + (length<0);                               % count epochs?!
```

- Machine learning models coded manually
- Increase of development time due to need of testing every model manually
- Difficult to track the behavior of the model

Classification Learner App



- Several Machine learning models immediately available (SVM, K-NNs, Bagged trees...)
- Effortless integration into already developed algorithm
- User friendly development tool

idneo

Summary

Development of an artificial vision system using MathWorks Tools:

- Use of Image Processing Toolbox to extract band features.
- Bands classification using machine learning (both custom built and Machine Learning toolbox)
- System performance evaluation with custom APP using MATLAB Compiler.
- Deploy image processing algorithm to an embedded system with MATLAB Coder and Embedded Coder.

Best Practices:

- Take profit of MATLAB tools to analyze data and develop the code. Time and resources dedication to what is worth instead of coding.
- Develop an APP to test an evaluate code performance, continuous integration of development & verification.
- Metrics evaluation with a big image dataset to quantify system performance.

Summary

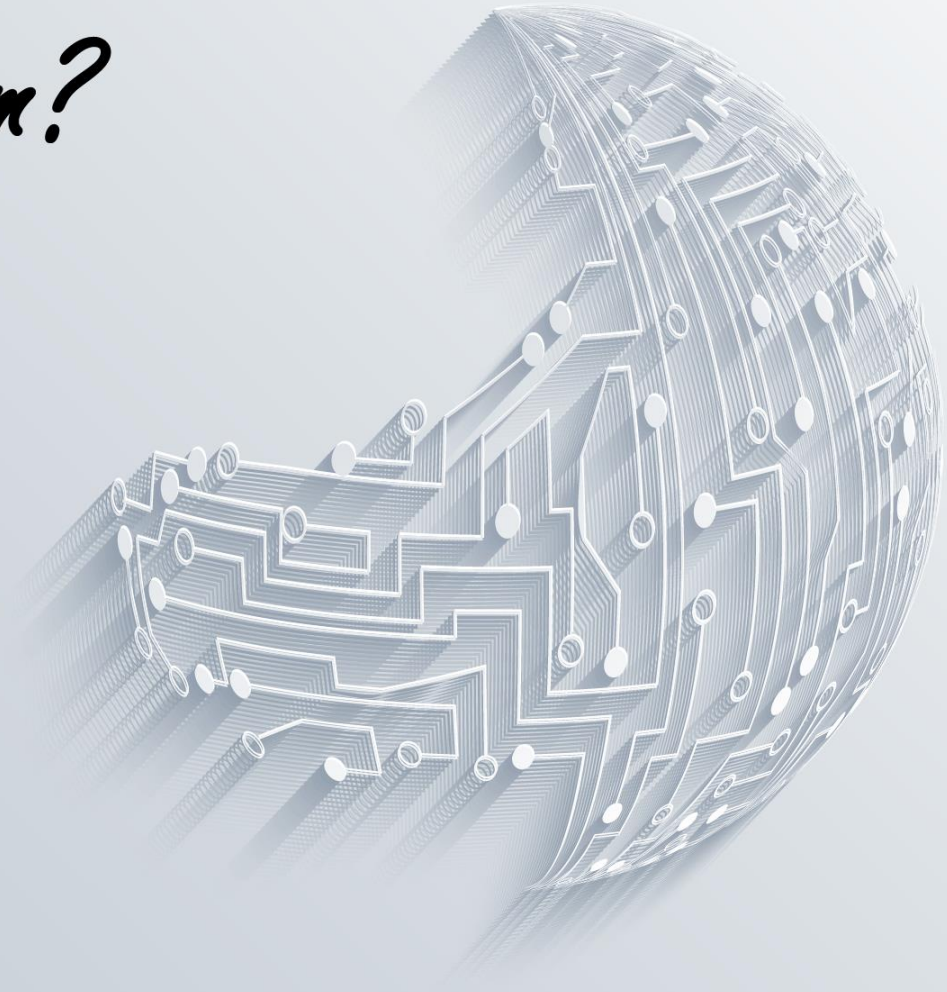
Recommendations:

- MATLAB Compiler.
- Classification Learner App.
- MATLAB Coder and Embedded Coder.

Future plans:

- Continuous improvement of algorithm accuracy. Final iteration planned.
- Implement the analysis of new card models.
- Explore possibilities of developing a Deep Learning framework for image classification in a strongly regulated market like medical devices.

What's your Dream?



Headquarters
Polígono Industrial Can Mitjans S/N
08232 Viladecavalls – Barcelona (Spain)
T (+34) 93 700 82 00
F (+34) 93 700 81 40

US Office
10900 North Tantau Ave
Suite 200
Cupertino, CA 95014 (US)

www.idneo.com

Idneo technologies

@Idneo_Tech