DataProphet Project 4

Sneha Iyer, Michael Lembck, Charles Yu,
Aggie Ly, William Parrish

Manager: Nelius Coomans

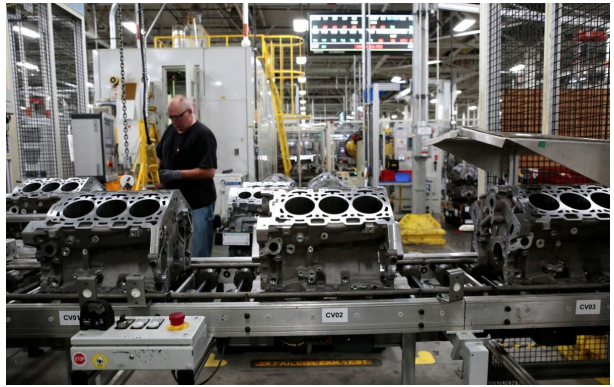
Background

dataprophet™
machine learning specialists

DataProphet is an AI-as-a-Service company of machine learning specialists with a global presence. They work with manufacturers worldwide to optimize their production KPIs via existing plant data and machinery. They specialize in optimizing the complex manufacturing processes of key industrial verticals with state-of-the-art machine learning. Their AI-driven solutions leverage the existing data streams from clients' plant's production line equipment to identify process efficiencies and faulty machine behaviour.

Overview

- Task: help solve a traceability problem at a large manufacturing plant
  - capture the inspection results per product unit (engine blocks).
- 3 main tasks:
  - Coding a pipeline for a vision system
  - Image labeling
  - Building and training OCR models in order to read product unit details from the product



Cameras

- There are three cameras, each to capture a different part of the engine block
  - Mould Number - format is 3 or 4 digit number (zero padded)
  - Cast Date - mix of chars and numbers
  - Block Type - mix of chars and numbers

How it works

- Trigger, the triggering is induced by an external impulse
  - Sensors detect engine block movement on an assembly line
- Sensors will trigger the input and previously programmed Programmable Logic Controller will trigger the output which is the camera and the camera will automatically starts the image capturing/recording
- New image acquisition is started, image is captured and sent to PC/stored in folder
- Programming the image capturing script:
  - Spinnaker API
  - Trigger Delay: Specifies a delay, in milliseconds (0 to 10,000), between the time the trigger is received and the time the vision system (camera) begins the acquisition.
  - Sleep times in between cameras
  - Exposure times of cameras

Image Pipeline

- Two main parts: trigger monitor and capture → 2 service scripts that when started, executed 2 python scripts trigger_monitor.py (given) and IX python script (our task)
- trigger_monitor.py:
  - Script that monitors a data acquisition system (DAQ) for a trigger signal from a programmable logic controller (PLC).
- Capture Task
  - trigger() - started camera acquisition
  - fetch_image() - called GetNextImage(), checked to make sure image was not incomplete

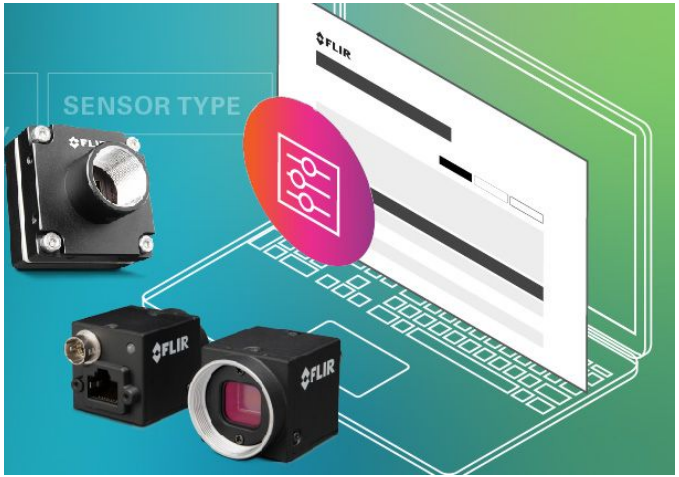


Image Capture Attempts

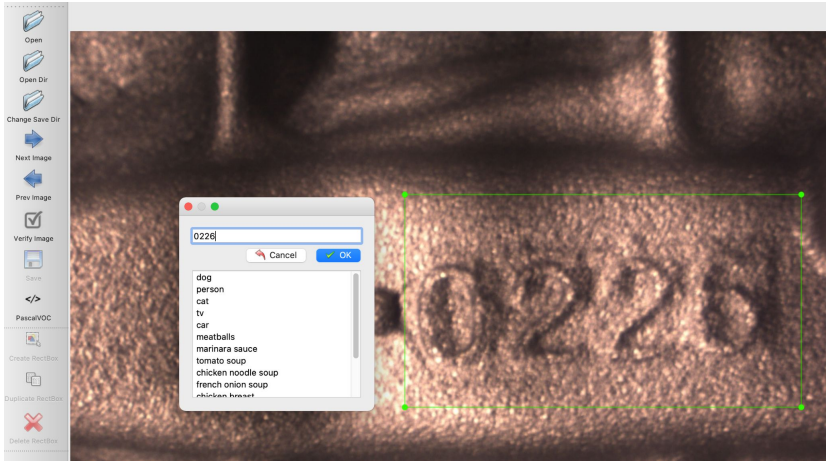
- Call trigger() for all cameras and fetch_image() for all cameras in two separate for loops with sleep() times in between to capture images
  - Fractured images, play around with sleep times in order to try and get clean images
- single_capture() which called trigger() and fetch_image() for one camera
  - Fractured images, buffer error
- video_capture() which got frames from a video and took one frame as the "image"
  - Good images, but lower quality
- hdr (high dynamic range) capture which took multiple images and meshed them together to create a composite image
  - Good images and higher quality -- help with modeling

Problems/Errors

- Errors with reference to camera, buffer, memory, etc
- Trial/Error with different image capture methods took time
- Possible hardware issues
  - Found that the 3rd camera was not working
- Problems with data acquisition, using cameras in succession (in a loop)
  - Ended up using only 1 camera - mould number
- Images end up with only background walls (no block) in the frame

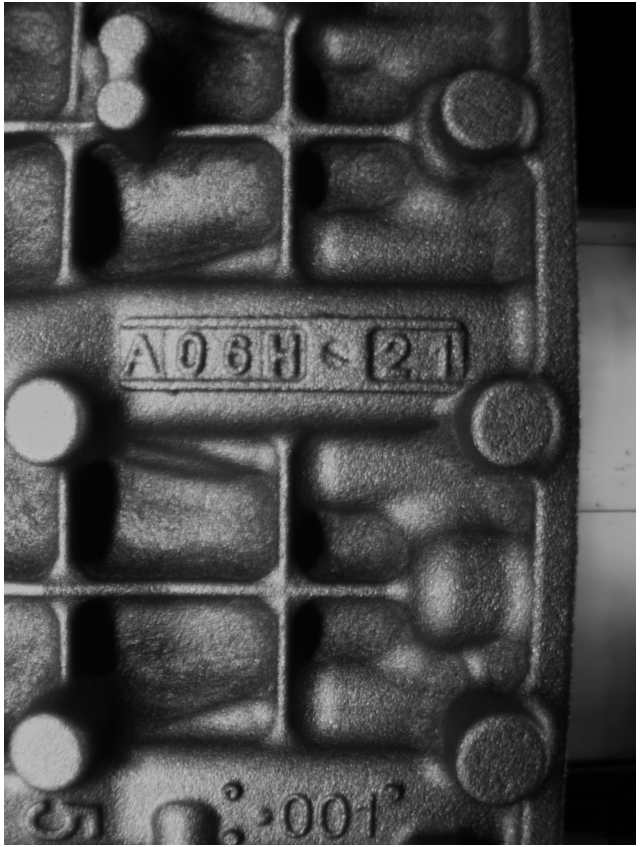
Image Labeling

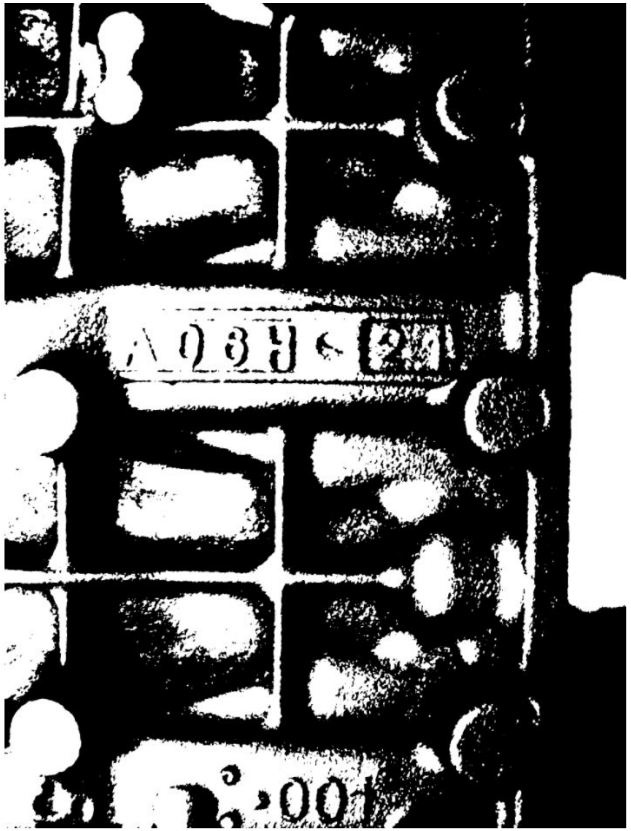
- Script utilized video_capture() method to take and store images
- Used LabelImg - a graphical image annotation tool
- Annotations are saved as XML files in PASCAL VOC format
- ~1000 images total

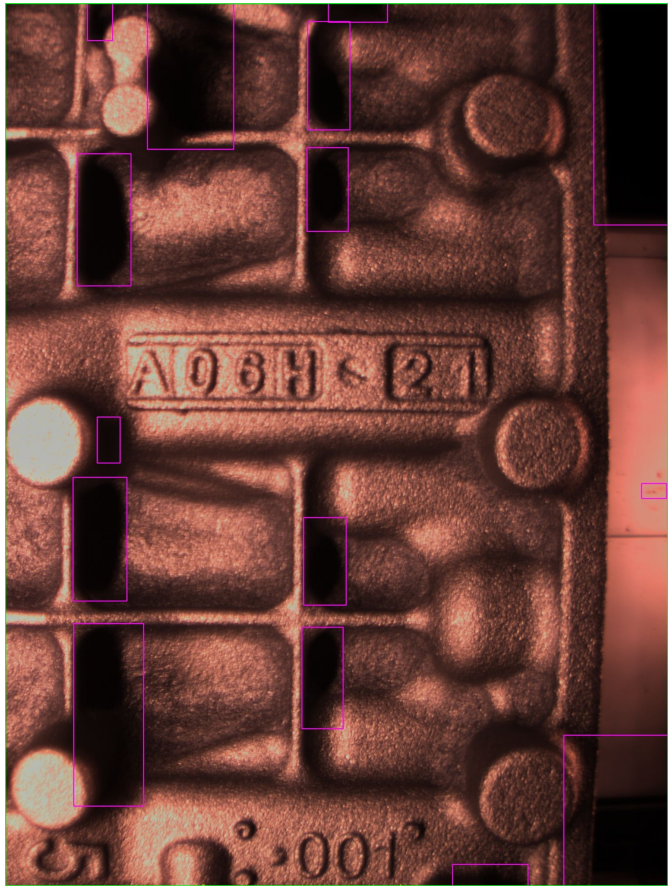


Region Detection

- Region detection to see where text is
- Homography to get text in legible format
  - Rotating 90 degrees
- Filtering in terms of pixel values to get high contrast sharp image
  - Many different image processing algorithms, OpenCV
- Crop the image to get only the region we wanted
- Goal: pass the processed image into ocr character recognition model for supervised learning


A06H-21


A06H-21


A06H
2.1
001


A06H
2.1
001

Supervised Learning Model

- Image Classification Model
- Attempted to isolate each digit by cropping at general x,y coordinates
  - 3-4 digit codes located in different positions → difficult to isolate each digit
  - Now we are checking consistency of x-coordinates
- Attempted to isolate each digit with region detection/contours
  - Contrast was not good enough
- Images: jpg files, Labels: 3-4 digit codes stored in xml file
  - Wrote a script to convert xml to csv to txt
- Scikit learn library
  - SVC: Support Vector Classification
  - General image classification model
  - Working on classifying each digit instead of entire code

Discussion/Future Work

- Coding an image pipeline
  - Working with Spinnaker API, camera software/hardware, python/terminal/service scripts
- Image processing, region detection
  - Image labeling software
  - OpenCV
- Supervised learning
  - Scikit learn
  - OCR models
- Future work
  - Get consistent images with trigger signal
  - Make a region detection model → higher accuracy
  - Better image processing, high contrast images → better results
  - Currently only 1 camera → 3 camera system

Thank you!