Creating a pick and place solution for 3D printers

Bachelor Thesis

Felix Kolwa

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics
Technical Aspects of Multimodal Systems

2.7.2019

Overview

1. Motivation
2. CV based PNP
3. CV Pipeline
4. Testing
5. Outlook

Motivation

- 3D printers are getting more popular and affordable
- Great similarities between 3 axis 3D printers and PNP machines
- A lack of established soft- and hardware to combine both

SMT PNP Machines

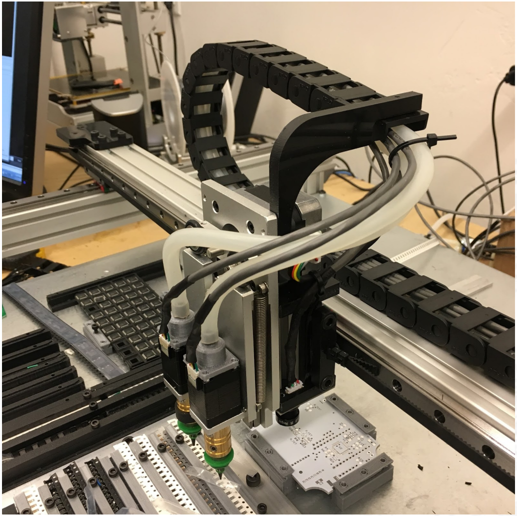


Figure: DIY PNP machine designed by anthony.webb. [hac19]

SMT PNP Machines

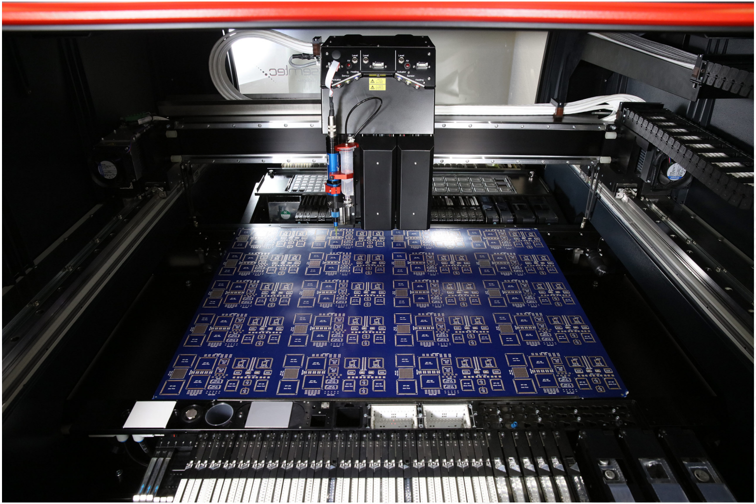


Figure: Commercial Pick and Place Machine by Essemtec. [ess19]

A Computer Vision based approach

Requirements

- Cameras to see the components
- A gripper to pick and place the components
- A tray to hold the components

Hardware Setup

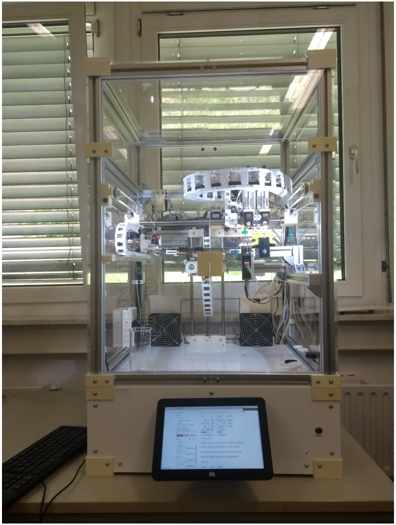


Figure: Modified Kühling&Kühling Reprap 3D printer.

Hardware Setup

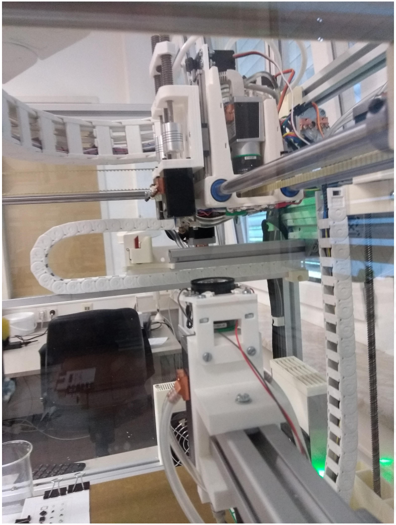


Figure: The cameras.

Hardware Setup

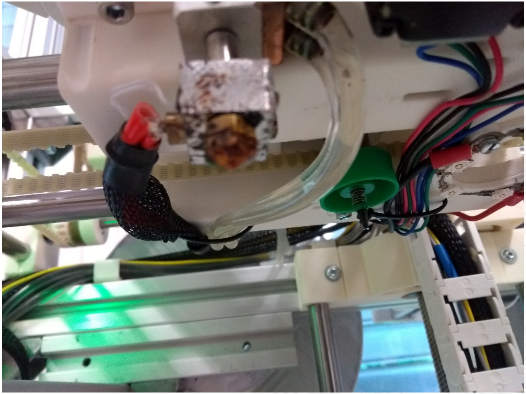


Figure: The vacuum gripper.

Hardware Setup

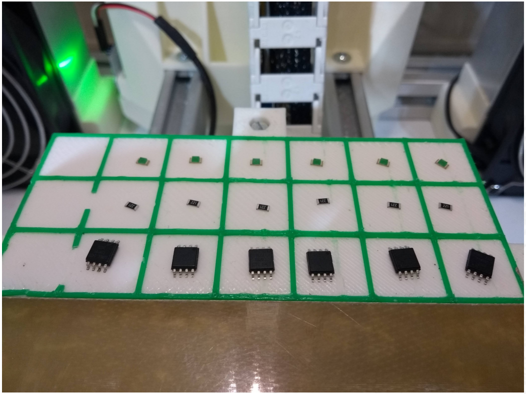
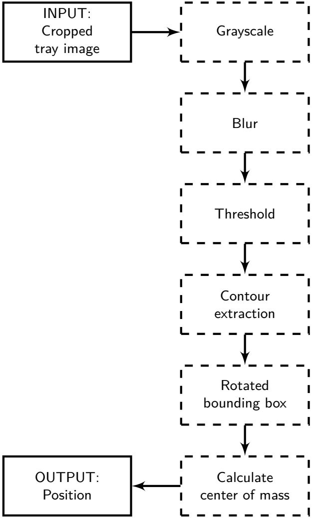
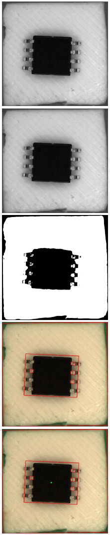


Figure: The component tray.

The existing Computer Vision Pipeline

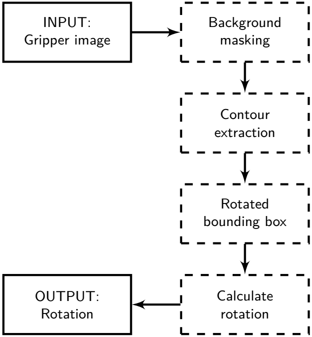


(a) Original cv pipeline for picking

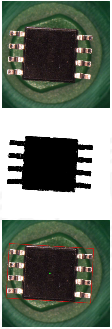


(b) Output images

The existing Computer Vision Pipeline

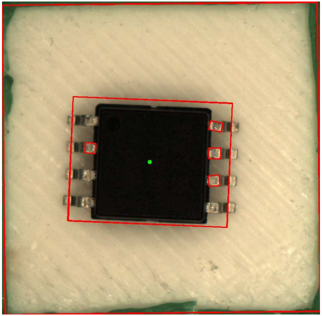


(a) Original cv pipeline for placing

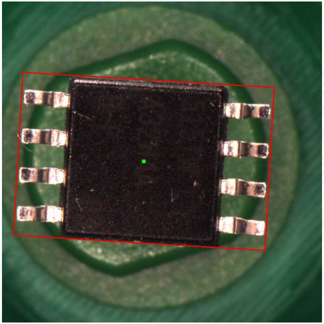


(b) Output images

Pipeline output: Good case

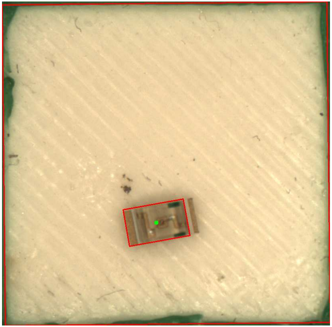


(a) Detected position

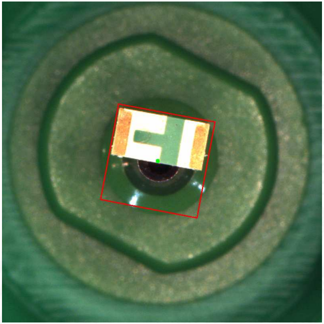


(b) Detected rotation

Pipeline output: Bad case



(a) Detected position



(b) Detected rotation

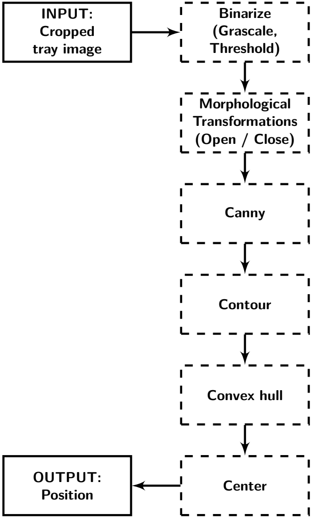
Room for improvement

- Dependent on manual configuration
  - Threshold values
  - Color range for background masking
  - Conditions change with components
- Imprecise bounding rectangles
  - Position offsets
  - Rotation offsets

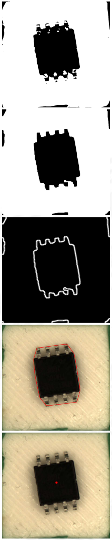
Goals

- Improve reliability
  - Automate manually configured values
- Improve precision
  - Replace bounding rectangles
  - Implement shape detection

Picking

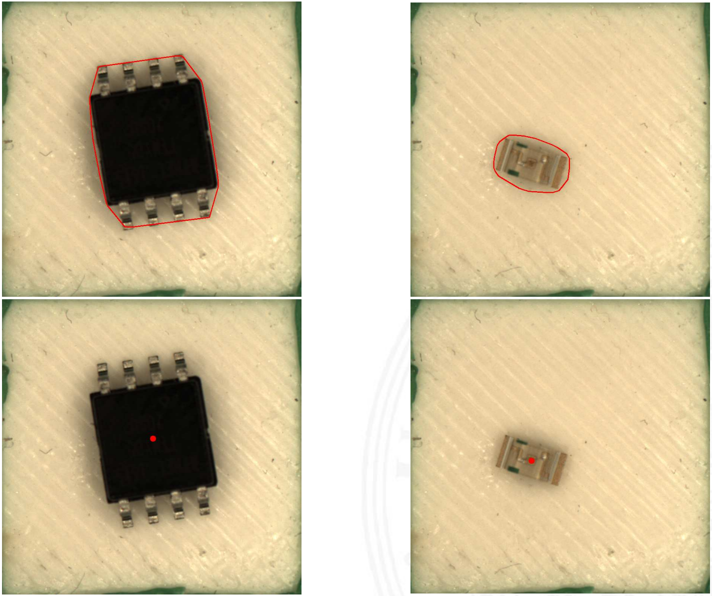


(a) New cv pipeline for picking



(b) Output images

Picking: Example output



Template matching

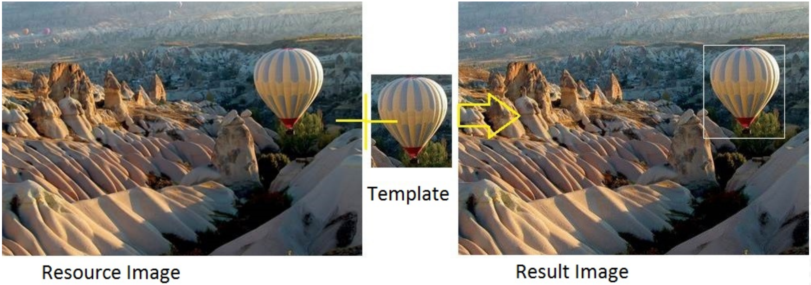


Figure: Simple template matching [tem19]

Pro:

- Easy to Implement
- Fast

Contra:

- No scaling or rotation
- Needs precise image data for successful matching

RANSAC

RANdom **SA**mple **C**onsensus

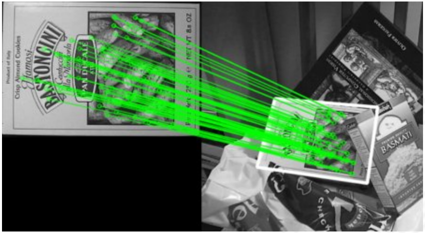


Figure: Feature Matching+Homography [ope19]

Pro:

- Well documented
- Efficient for calculating homography between two images

Contra:

- Most implementations relying on *non-free* algorithms

Generalized Hough Transform

Pro:

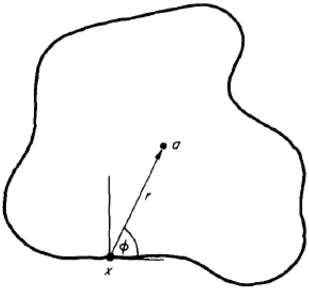
- Robust to partial or slightly deformed shapes
- Robust to the presence of additional structures
- Tolerant to noise
- Can find multiple occurences of a shape during the same processing pass

Contra:

- Computation intensive
- Memory intensive
- Slow

Building the rTable

INPUT: binary template → Detect edges → Choose reference point → Calculate gradient orientation ϕ → Calculate displacement vector r → Store r as function of ϕ → **OUTPUT:** rTable



displacement vector r for an arbitrary boundary point x [Bal81]

Generalized Hough Transform

Localizing the object

INPUT: binary search image → Detect edges

↓

Calculate gradient orientation ϕ

↓

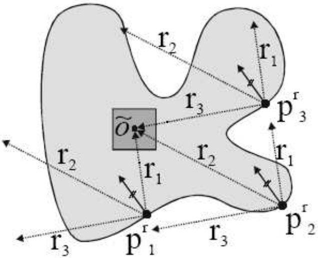
Retrieve displacement vector r from rTable

↓

Compute position of reference point

↓

Vote for this point → **OUTPUT:** Matrix containing possible candidates



Reconstruction of the reference origin [MCA03]

Picking: Example output

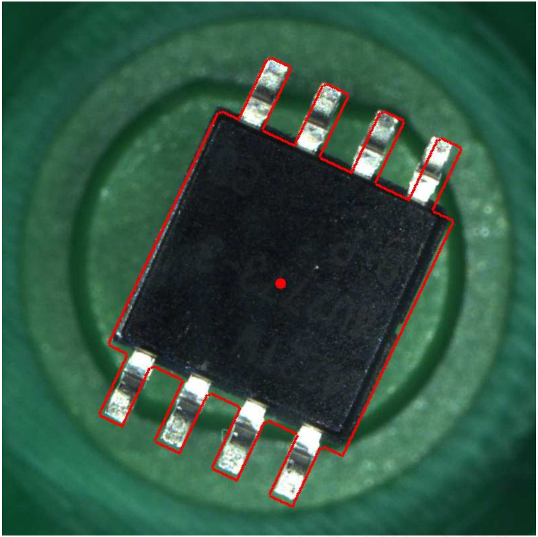


Figure: Matched template

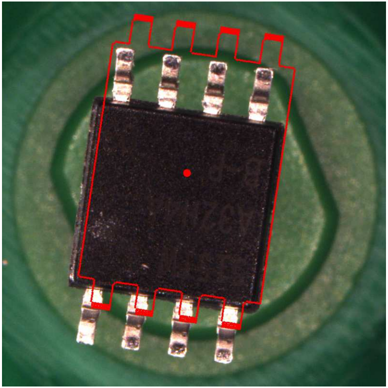
Picking: Template source

```
<part id="2" name="ATTiny2">
  <position box="2"/>
  <size height="1.87" width="5.38"/>
  <shape>
    <point x="-2.6" y="-2.6"/>
    <point x="-2.6" y="2.6"/>
    <point x="2.6" y="2.6"/>
    <point x="2.6" y="-2.6"/>
  </shape>
  <pads>
    <pad x1="-2.155" y1="-4.0" x2="-1.655" y2="-2.054"/>
    <pad x1="-0.895" y1="-4.0" x2="-0.395" y2="-2.054"/>
    <pad x1="0.375" y1="-4.0" x2="0.875" y2="-2.054"/>
    <pad x1="1.645" y1="-4.0" x2="2.145" y2="-2.054"/>
    <pad x1="-2.155" y1="2.054" x2="-1.655" y2="4.0"/>
    <pad x1="-0.885" y1="2.054" x2="-0.385" y2="4.0"/>
    <pad x1="0.385" y1="2.054" x2="0.885" y2="4.0"/>
    <pad x1="1.655" y1="2.054" x2="2.155" y2="4.0"/>
  </pads>
  <destination x="20" y="10" z="0" orientation="45"/>
</part>
```

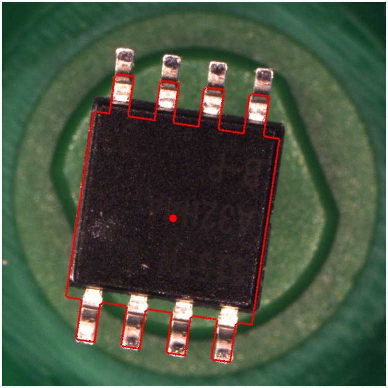
(a) G-Code

(b) Template output

Picking: Example output



(a) Matched template exported from Eagle CAD



(b) Matched template based on rough approximation

The idea

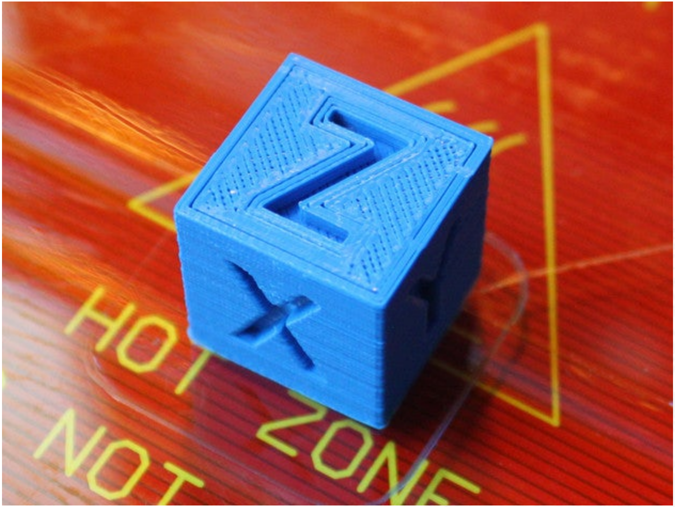


Figure: A 3D printed calibration cube [thi19]

The template

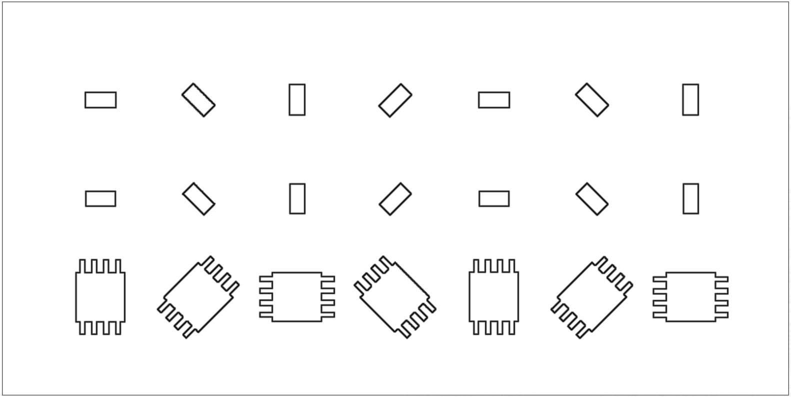


Figure: The paper template

The template

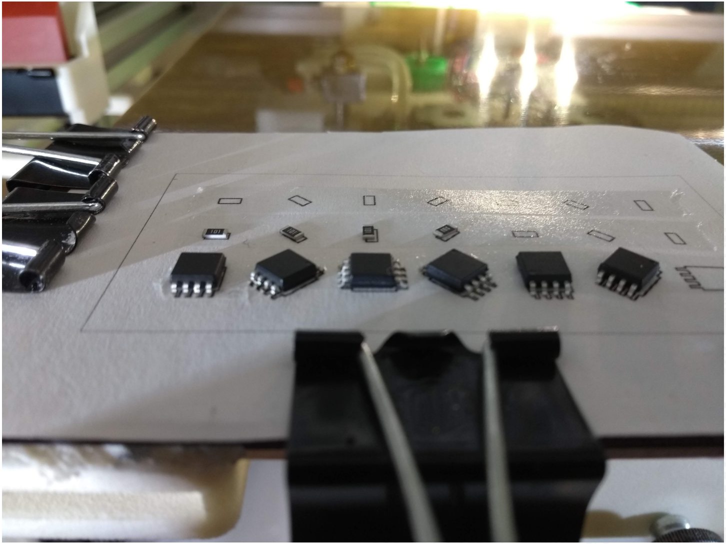


Figure: The paper template attached to the print bed

The components

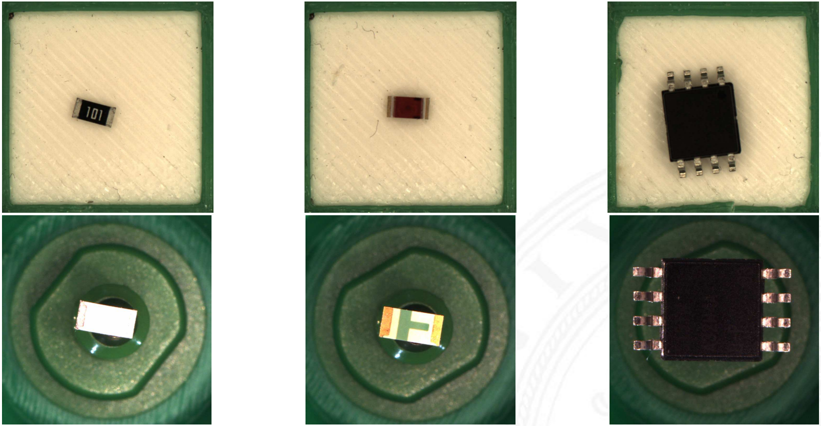


Figure: The chosen components

The procedure

Test structure

1. Attach printed paper template to print bed
2. Apply adhesive
3. Run print job with provided G-Code file
4. Place parts
5. Take pictures of placed parts
6. Document runtime

Running the test

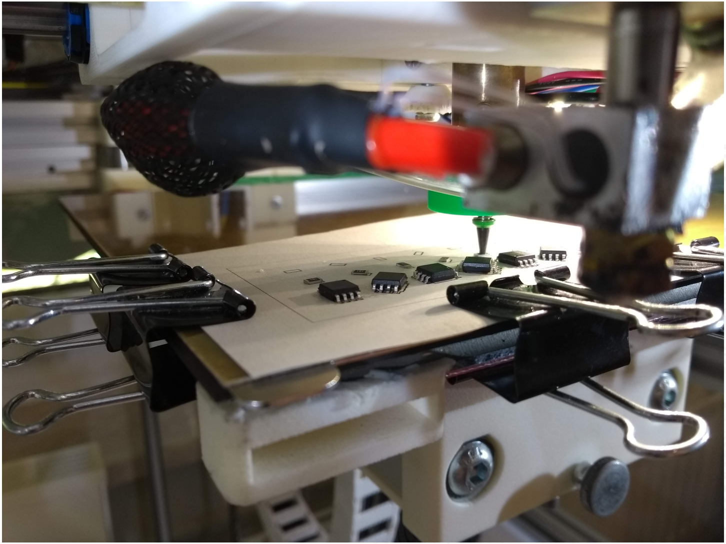
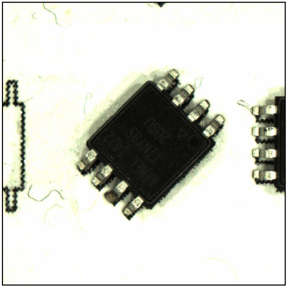
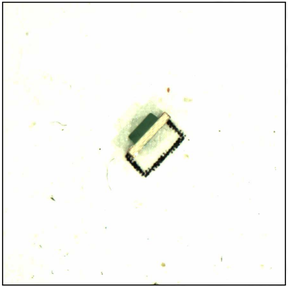


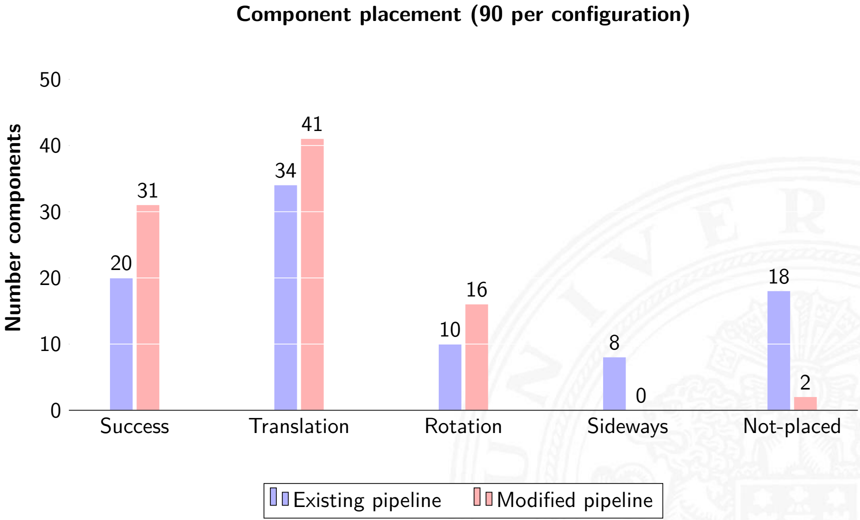
Figure: A PNP test in progress

First results





First results: Combined



Average time per component

Existing pipeline: 10.4658s Modified pipeline: 10.4475s

Common failures

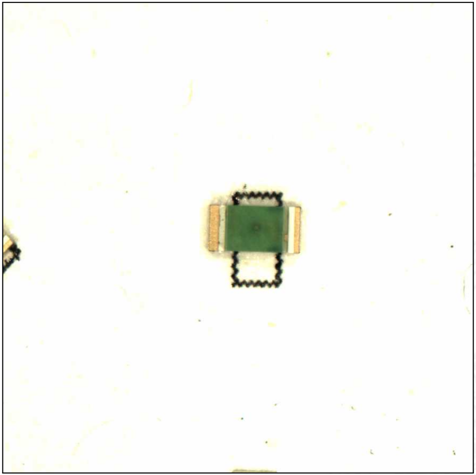
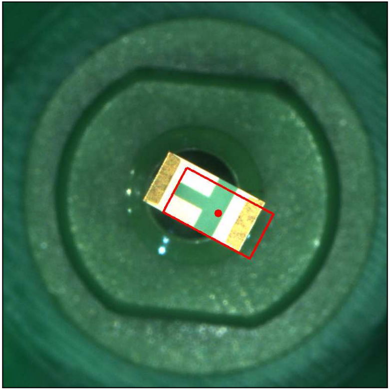


Figure: 90 degree rotation offset

Common failures



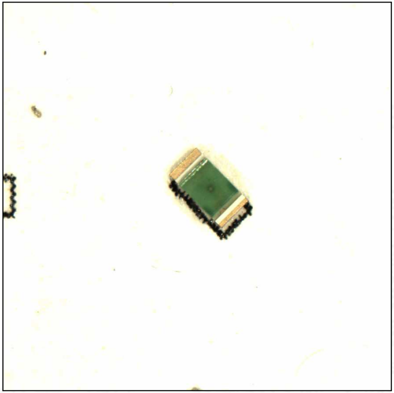


Figure: Bad template matching

Common failures

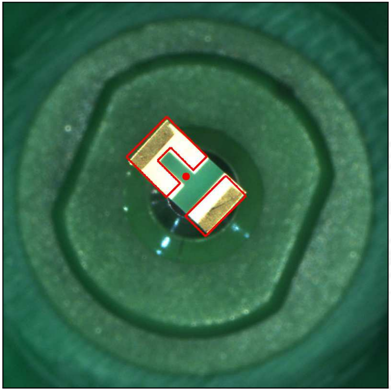


Figure: Modified template

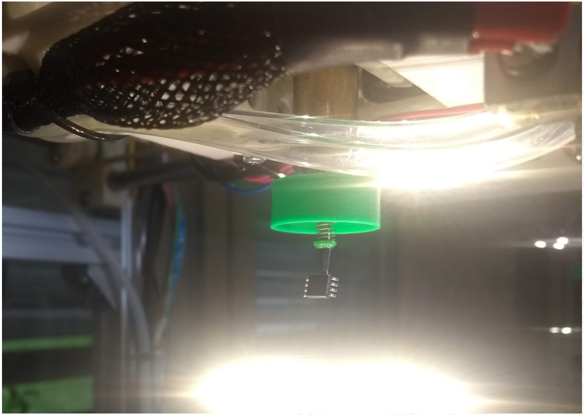
Outlook

Further planning

- Improve template matching
- More precise templates
- Continue testing

Thanks

Thank you for your time!



Questions?

[Bal81] D. H. Ballard., *Generalizing the hough transform to detect arbitrary shapes.*, Pattern Recognition, 1981, pp. 13(2):111–122.

[cha17] *Visual servoing based object pick and place manipulation system.*, Shawn X. Wang (editor) Current Trends in Computer Science and Mechanical Automation Vol.2: Selected Papers from CSMA2016, 2017, pp. 334—-341.

[ess19] *Essemtec*, `https://essemtec.com/en/products/pick-and-place/puma/`, Accessed: 01.07.2019.

[hac19] *A diy pnp machine designed by anthony webb*, `https://hackaday.io/project/9319-diy-pick-and-place`, Accessed: 01.07.2019.

References (cont.)

[MCA03] Ulrich M., Steger C., and Baumgartner. A., *Real-time object recognition using a modified generalized hough transform.*, Pattern Recognition, 2003, pp. 36(11):2557–2570.

[oct19a] *Octopnp - octoprint plugin for camera based pick 'n place operations*, `https://github.com/platsch/OctoPNP`, Accessed: 01.07.2019.

[oct19b] *Octoprint - web interface for your 3d printer*, `https://octoprint.org/`, Accessed: 01.07.2019.

[ope19] *Opencv documentation about ransac*, `https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_feature_homography/py_feature_homography.html`, Accessed: 01.07.2019.

References (cont.)

[tem19] *Blog about template matching*, https://riptutorial.com/opencv/example/22915/template-matching-with-java, Accessed: 01.07.2019.

[thi19] *Calibration cube by idig3dprinting. thingiverse*, https://www.thingiverse.com/thing:1278865, Accessed: 01.07.2019.

[Was15] Florens Wasserfall, *Embedding of smd populated circuits into fdm printed objects.*, Proceedings of the 26th International Solid Freeform Fabrication Symposium, 2015, pp. 180–189.