PCB part recognition for material recycling

BY

Bernhard Föllmer

Department of Machine Tools and Factory Management

Chair of Industrial Information Technology

Submitted in partial fulfillment of the requirements

for the degree of

Master of Science in

Physical Engineering Science

at the

Technical University of Berlin

13.10.2014

Abstract

Sadsad

Table of Contents

Table of Contents

List of figures

List of tables

List of abbreviations

DFT
Discrete fourier transform, 32
FFT
Fast fourier transform, 33
LoG
Laplacian of Gaussion, 82

1. Introduction

Ewt

1.1 Background

Ads

- Industrial PCB recycling process chain
- INPIKO erklären

1.2 Problem formulation

The production of electric and electronic equipment (EEE) is increasing worldwide. At the end of the life the equipment ends up as waste electric and electronic waste (WEEE). This development requires an End-of-life management system which serves the following goals:

- Reduction of materials going to landfill, and minimization od landfill-volumes
- Recycling of materials in order to keep the maximum economic and environmental value and to avoid new material extraction
- Reduction of emissions of environmentally relevant substances, for example through leaching from landfill sites, incineration slags and off-gasses from combustion processes

Huismann et al. 2004)

Recycling of WEEE is an important subject not only from the point of waste treatment but also from the recovery of valuable materials and the reuse of electronic components. WEEE is diverse and complex in terms of materials and components makeup as well as the original equipment's manufacturing processes. Electronic products, in particular IT and communication equipment contains a lot of precious metals (gold, silver, palladium) and special metals (indium, selenium, tellurium, tantalum, bismuth, antimony). The precious metals are mainly found in printed circuit boards (PCBs). The concentration of precious metals in PCBs is usually much

higher than the concentration in ores, especially for gold and palladium. Moreover the extraction of precious metals through mining is associated with negative environmental impacts through significant emissions of greenhouse gases and energy, water and land usage. Moreover the high economic value of precious metals on the word market as well as the limited available reserves of precious metals requires an improvement of recovering precious metals from WEEE. The proportion of PCBs in WEEE over different equipment type is around 9% (Chancerell 2009). The concentration of precious metals in unshredded printed circuit boards is around 669 g/t of silver, 135 g/t of gold and 50 g/t of Palladium.

- Reuse???

A simplified recycling chain for WEEE is shown in Figure 1. The recycling chain consist of three steps. The first step is the Collection of WEEE which is out of focus of the improvement of the recycling chain in this thesis. The pre-processing step consists of manual sorting and dismantling as well of shredding and automated sorting. The improvement of the pre-processing stage is the main focus of this thesis

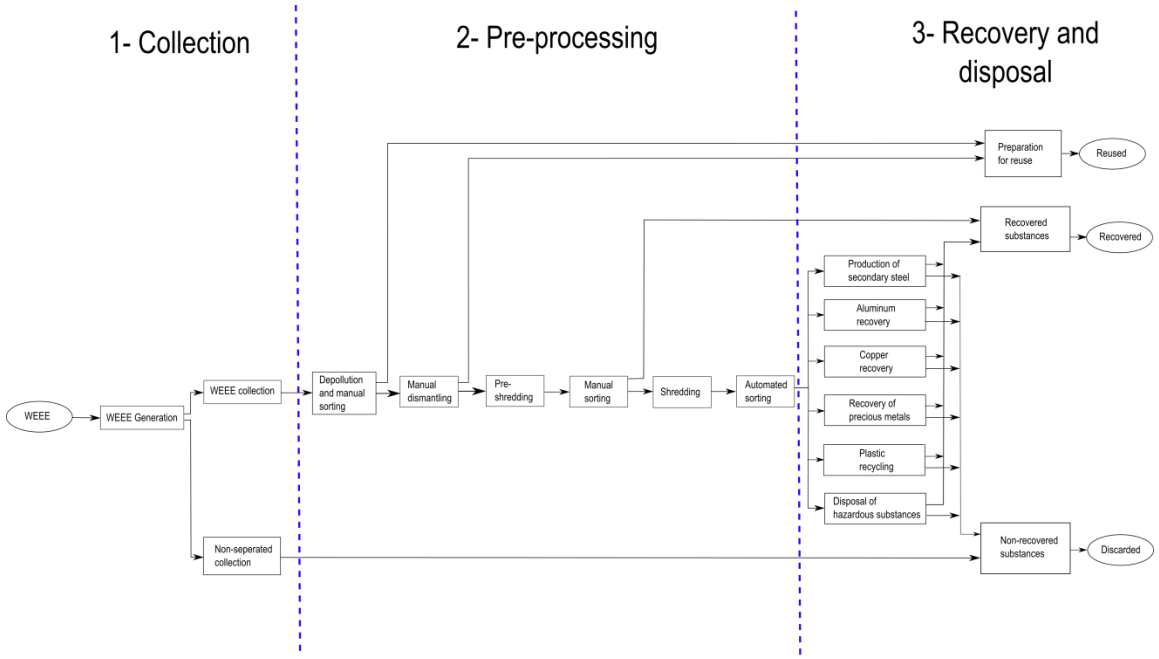


Figure 1: Simplified recycling chain for WEEE

The mass balance of the preprocessing step is shown in Figure 2.

- PCB anteil (9%)?

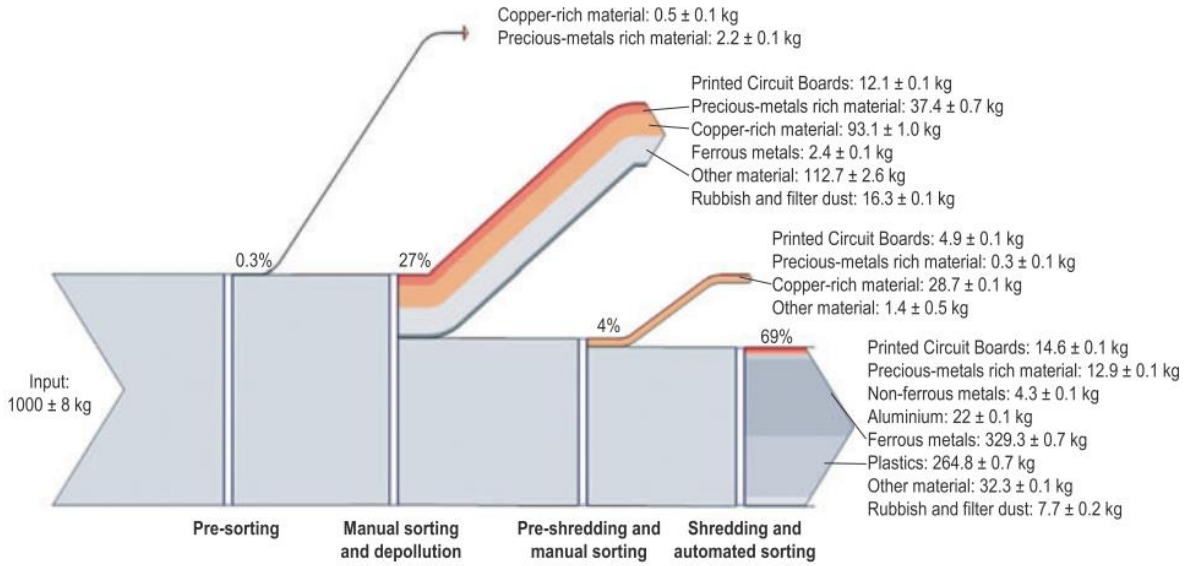


Figure 2: Mass balance of the preprocessing of 1,000 kg of input WEEE (Chancerell 2009)

A comparison of the input concentration and the output concentration of precious metals shows that only about a quarter of the gold and palladium and about on tenth of silver are send to the output fraction from which precious metals will be directly recovered. Most of the precious metals go to the most mass relevant fractions. Per ton of input WEEE the company operating the facility does not get any revenues for around 16.5 g gold and 5.3g palladium. At a price of $900 per ounce of gold and $370 per ounce of palladium (average price for 2008 [UGS 2009]), this means that a metal value of $524 for gold and almost $70 for palladium per ton of treated WEEE is lost. More shredding results in a decrease of concentration of precious metals in PCBs (Chancerell 2009). To reduce the losses of precious metals in preprocessing, in particular during shredding and subsequent sorting, the first and most straight-forward approach is to reduce the quantity of precious metals entering in the shredder. This implied

adjusting the manual sorting step at the beginning of the process to remove most precious metal-rich materials. This requires knowledge about the location of precious metals in WEEE, which is currently partially missing (Chancerell 2009). Characterization of the waste stream is of paramount importance for developing a cost-effective and environmentally friendly recycling system (Jirang Cui 2003).

1.3 Purpose

The purpose of this work is to improve the preprocessing step of the recycling chain by an improved automatic characterization of the PCB waste stream. The automatic characterization of PCB waste is done on component level. The unshredded or pre-shredded PCBs are inspected by an automatic optical inspection system (AOI-System) based on an electronic component recognition database which contains information for the electronic component recognition system.

Information about the content of valuable materials (gold, silver, palladium,…) or hazards materials (heavy metals, brominated flame,…) are used to automatically generate PCB composition model which contains the location and quantity of specific materials depending on the electronic components of the PCB. This model can help for automatic or manual selective disassembly of precious metal rich components or hazard material rich components.

Information about the economic value of reusable electronic components helps to locate reusable components from an economic point of view. The increase of reuse rate decreases the negative environmental impacts caused by the production of new electronic components and increases the revenue of recycling companies.

An improved recycling chain model with the approach examined in this work is shown in chapter 7.1.

- continue

- Prozesskette optimieren

2. Background Theories and related works

2.1 Data fusion

2.2 Feature extraction algorithms

2.2.1 A priori knowledge generation

2.2.2 Single seed region growing approach for color images

For background segmentation and feature extraction from electronic part segments a region growing approach is used for region segmentation. The region growing approach is a pixel based image segmentation method since it involves the selection of initial seed pixel. The region growing algorithm examines neighboring pixel of a region or the initial seed pixel and determines if the neighboring pixel should be added to the region (Wikipedia 2014). The first step is the selection of seed point (x, y). The seed point selection is depending on the segmentation goal and based on user criterion. The seed point selection is explained in detail for the specific methods (3.3.3 Segment based feature extraction, 3.2.2 Color based PCB surface detection). The seed pixel is the first region, from which neighboring pixel are added to grow the region iterative depending on a region membership criterion. In this approach the region

growing segmentation is used to segment color images. The criterion to add adjacent pixel $f(x,y)$ to the region pixel PG is the Euclidian distance $DIST$ between the color of the adjacent pixel and the mean color value of the region PG_{mean}. Before segmentation, the image was converted from RGB color space to HSV color space and the gray scaled values in the three channels where linear scaled between 0 and 1.

$$DIST = \sqrt{D_H + D_S + D_V} \tag{1}$$

$$D_H = (f(x,y,1) - PG_{mean}(1))^2 \tag{2}$$

$$D_S = (f(x+1,x+j,2) - PG_{mean}(2))^2 \tag{3}$$

$$D_V = (f(x+1,x+j,3) - PG_{mean}(3))^2 \tag{4}$$

$$PG_{mean}(1) = \frac{1}{\#PG}\sum_i f(PG(i),1) \tag{5}$$

$$PG_{mean}(2) = \frac{1}{\#PG}\sum_i f(PG(i),2) \tag{6}$$

$$PG_{mean}(3) = \frac{1}{\#PG}\sum_i f(PG(i),3) \tag{7}$$

Is the distance smaller than a determined threshold $THR = 0.02$, the pixel is added to the region. If the distance exceeds the threshold, the pixel is not added to the region. If the distance from all neighboring pixel to the region exceed the threshold, the region growing stops and the segmented region is determined as a segment of the image (O. Verma 2011). The pseudo code of the single seed region growing approach is shown in Code 1.

```
SEED: position of seed (x,y)
RCOUNT: Counter of keep track of current region being grown
PG – stack to store pixel to grow
BP – stack to store boundary pixels of grown region
REGION: matrix with same size if image I, storing the labels of growing region
CP(j): 4-neighbours of CP, j=1,2,3,4
```

PSEUDOCODE:

```
Region_Growing(HSV image I)
	THR=0.02
	SEED=(x,y)
	RCOUNT=1
	i=1
	j=1
	PG(i)=SEED
	While PG not empty
		CP=PG(i)
		i=i-1
		For(4-nb of CP, k=1:4)
			If(REGION (CP(k) not labeled)
				Calculate: DIST(SEED,CP(k))
				If(DIST<THR)
					REGION(CP(k))=1;
					i=i+1
					PG(i)=CP(k)
				Else
					j=j+1
					BP(j)=CP(k)
				End if
			End if
		End for
	End for
End
```

Code 1: Single seed region growing pseudo code

2.2.3 k-means clustering

In the color based PCB surface recognition algorithm in chapter 3.2.2, the k-means clustering algorithm is used to find clusters of PCB surface segments. The algorithm is an unsupervised procedure with the goal to find k mean vectors $\mu_1, \mu_2, \dots, \mu_k$ which represents the center of the k clusters. The k-means clustering is an iterative method where k is the number of clusters. The determination of the number of clusters is described in detail in the belonging chapter. In this approach the initial means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots \boldsymbol{\mu}_k$ where selected randomly from the sample space. The squared Euclidian distance $\left|\left|\boldsymbol{x}_k - \widehat{\boldsymbol{\mu}}_i\right|\right|^2$ is computed for each sample and the nearest mean $\widehat{\boldsymbol{\mu}}_m$ is selected to approximate $\widehat{\boldsymbol{P}}\left(w_i \middle| \boldsymbol{x}_k, \widehat{\Theta}\right)$ as:

$$\widehat{\boldsymbol{P}}\left(w_i \middle| \boldsymbol{x}_k, \widehat{\Theta}\right) \simeq \begin{cases} 1 & if\ i = m \\ 0 & otherwise \end{cases} \tag{8}$$

After approximating $\widehat{\boldsymbol{P}}\left(w_i \middle| \boldsymbol{x}_k, \widehat{\Theta}\right)$ the means $\widehat{\boldsymbol{\mu}}_1, \widehat{\boldsymbol{\mu}}_2, \dots \widehat{\boldsymbol{\mu}}_k$ are recomputed by:

$$\widehat{\boldsymbol{\mu}}_i = \frac{\sum_{k=1}^{n} \widehat{\boldsymbol{P}}\left(w_i \middle| \boldsymbol{x}_k, \widehat{\Theta}\right) \boldsymbol{x}_k}{\sum_{k=1}^{n} \widehat{\boldsymbol{P}}\left(w_i \middle| \boldsymbol{x}_k, \widehat{\Theta}\right)} \tag{9}$$

The approximations of $\widehat{\boldsymbol{P}}\left(w_i \middle| \boldsymbol{x}_k, \widehat{\Theta}\right)$ and the recomputations of the means are repeated until the approximations do not change compared to the previous iteration step (Duda 2001). The pseudo code is shown in Code 2.

```
PSEUDOCODE:

k_Means_Clustering(samples)
begin initialize n, k, μ1, μ2, …, μk
       do classify n samples according to nearest μi
              recompute μi
       until no change in μi
   return μ1, μ2, …, μk
end
```

Code 2_ k-means clustering pseudo code (Duda 2001)

2.2.4 Normalized cross correlation for 2-D pattern matching

2.2.5 Image reconstruction with PCA

A set of m part images I_i each of size r x c is reshaped to a vectors $\boldsymbol{v_i}$ of size r*c x 1. First the mean vector $\boldsymbol{\mu}$ and the covariance matrix **C** are computed for all vectors according to (10) and (11).

$$\boldsymbol{\mu} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{v_i} \tag{10}$$

$$\mathbf{C} = \sum_{i=1}^{m} (\mathbf{v_i} - \boldsymbol{\mu})(\mathbf{v_i} - \boldsymbol{\mu})^T \tag{11}$$

Next the eigenvectors and eigenvalues are computed and sorted according to decreasing eigenvalues. This computation can be done in several ways in which Matlab implementation based on the QZ algorithm was used in this approach. The eigenvectors $\boldsymbol{e_i}$ with the k largest eigenvalues λ_i of the covariance matrix are used to construct the projection matrix **P** of size r*c x k. The projection of an image vector v_i into the eigenspace is given by

$$\boldsymbol{p} = \boldsymbol{P}(\boldsymbol{v_i} - \boldsymbol{\mu}) \tag{12}$$

The reconstruction of an image projects the image into the PCs and from this projection, try to recover the original image by applying the invers projection matrix. The projection and recover step is shown in whereas $\boldsymbol{v_i}'$ is the reconstructed image of the image $\boldsymbol{v_i}$.

$$\boldsymbol{v_i'} = \boldsymbol{P^T p} + \boldsymbol{\mu} = \boldsymbol{P^T P}(\boldsymbol{u} - \boldsymbol{\mu}) + \boldsymbol{\mu} \tag{13}$$

The reconstruction error is defined by the euclidean distance between the image $\boldsymbol{v_i}$ and its reconstructed image $\boldsymbol{v_i}'$.

$$d = |\boldsymbol{v_i} - \boldsymbol{v_i}'| = \sqrt{\sum (\boldsymbol{v_i} - \boldsymbol{v_i}')^2} \tag{14}$$

Often there will be just a view large eigenvalues whose eigenvectors contain the most information while the rest of the dimensions generally contain noise (Duda 2001).

2.3 Featureselection

Asd

- Wrapper, embedded
- Guyon
- Redundant features
- Combination of features

2.3.1 Fisher score

Fisher score is a variable ranking method that rates the efficient for discriminations for each feature. It can be applied in two-class problems as well as in multi-class problems. The score evaluates each feature by the ration of the between class variance to the within-class variance (Guyen 2003). Suppose we have a set of d-dimensional samples $x_1, \dots, x_n$, n_k is the number of samples in the subset D_k labeled ω_k and c is the number of classes. The Fisher score of the j-th feature is computed in (15).

$$F(x^j) = \frac{\sum_{k=1}^{c} n_k (\mu_k^j - \mu^j)^2}{(\sigma^j)^2}) \tag{15}$$

Where σ^j is the standard deviation and μ^j the mean of the whole data set corresponding to the j-th feature and x_i^j is the j-th feature of the sample $\boldsymbol{x}_i$.

$$(\sigma^j)^2 = \sum_{k=1}^{c} n_k (\sigma_k^j)^2 \tag{16}$$

$$\sigma_k^j = \sum_{\boldsymbol{x}_i \in D_k} x_i^j - \tilde{\mu}_k^j \tag{17}$$

$$\tilde{\mu}_k^j = \frac{1}{n_i} \sum_{\boldsymbol{x}_i \in D_k} x_i^j \tag{18}$$

$$\mu^j = \frac{1}{n}\sum_{k=1}^{c} n_k \tilde{\mu}_k^j \tag{19}$$

After computing the fisher score for each feature, it selects the top-m features as the subset of features. The number of features m can be fixed or depend on a score threshold. The score of each feature is computed independently of all other features. Therefore the feature subset can be suboptimal because features with low individual scores but a very high score when they are combined are discarded furthermore redundant features are not discarded (Q. Gu, Z. Li, J. Han 2012). In this approach the fisher score is only used in the two stage feature selection and not applied alone for feature selection (see chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**).

2.3.2 Random forest feature selection

The Random forest feature selection is based on the out-of-bag (oob) error estimation. Each tree is constructed using different bootstrap samples from the data. A subset of samples is left out and not used to construct the k-th tree (oob-samples). Each sample that was left out to construct the tree is predicted by the k-th tree and compared to the true class of the sample. This is done with all trees of the random forest and the error over all trees and out-of-bag-samples are summed and divided by the number of out-of-bag-samples (Breiman, www.stat.berkeley.edu 2014).

In the Random forest feature selection approach the oob-error is estimated. Now the values of the m-th feature of the oob-samples are randomly permuted and the the new oob-error is estimated. Subtract the number of oob-errors made by the variable-m-permuted oob-samples from the number of oob-errors made by the untouched oob-samples. The average of this number over all trees in the forest is the raw importance score for variable m. This raw importance score is divided by the standard deviation to get the z-score which is used as the variable importance score (Breiman, Random Forests 2001).

- Redundante features

- Plot tantalum importance
- Missing values (median, proximities)

2.4 Object Classification

Dwq

2.5 Random forest classifier

2.5.1 Introduction to Ensemble classifiers

In supervised learning a supervisor (teacher) provides a category label for each pattern in a training set which also are referred to classes or labels. The classification of pattern is based on classification models (classifiers) which are learning the reclassified patterns of the training set. An algorithm which constructs the model is called inducer and an instance of an inducer for a specific training set is called a classifier. The idea behind an ensemble classifier is to weight several individual weak classifiers and combine them to form a strong inducer. It is well known that ensemble methods can improve the prediction performance (Rokach 2010).

2.5.2 Introduction to Random forest ensemble classifier

The random forest is an ensemble classifier where the individual classifiers are unpruned tree predictors. The training algorithm of random forest applies bagging (bootstrap aggregating) for tree learning.

- Cart
- breiman

2.5.3 Random forest training

Given a training set $\boldsymbol{X} = \boldsymbol{x_1}, \dots, \boldsymbol{x_n}$ with response $= \boldsymbol{y_1}, \dots \boldsymbol{y_n}$, bagging repeatedly selects bootstrap samples of the training set and fits trees to the samples. For each tree in the random forest classifier selects k random training samples $\boldsymbol{X_b}, \boldsymbol{Y_b}$ (bootstrap samples) from the training set and trains the bagging trees f_b on $\boldsymbol{X_b}$ and $\boldsymbol{Y_b}$. The optimal number of trees in the random forest depends on the size and structure of the data. In general a few hundred to several thousand trees are used whereas the generalization error for forests converges to a limit as the number of trees becomes large (Breiman, Random Forests 2001). In random forests at each candidate split a random subset of features is selected. Typically for a dataset with p features $\sqrt{p}$ features are used in each split (Random forest 2014).

2.5.4 Random forest prediction

The random forest prediction of a sample is done by predicting each trained tree in the random forest and averaging the prediction results over all trees. The output of the random forest can be normalized by the number of trees and interpreted as a soft-output probability. The prediction output is shown in (20) whereas B is the number of trees in the forest and $\widehat{f_b}$ the trained tree (Random forest 2014).

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} \widehat{f_b}(\boldsymbol{x}) \tag{20}$$

2.5.5 Out-of-bag (oob) estimation

To train a k-th tree a random subset of training samples X_b, Y_b is used to construct the tree whereas each tree uses different bootstrap samples. The samples that are not used to construct the k-th tree are predicted by the k-th tree to get a classification. The estimation is calld out-of-bag estimation. In this way, a test set classification is obtained for each case. At the end of the run, take j to be the class that got most of the votes every time case n was oob. The proportion of times that j is not equal to the true class of n averaged over all classes is the oob error estimate (Breiman, www.stat.berkeley.edu 2014).

- overestimate

2.6 Support vector machine classifier

Support vector machine (SVM) is a learning algorithm that analyzes data and recognizes patterns used for classification and regression analyses. Given a set of training samples, each marked with one of two classification categories an SVM model can be trained to assign new samples into one category or the other. In addition to performing linear classification, an SVM can efficiently perform a non-linear classification by using the so called kernel-trick. The kernel-trick is a mapping of the input data to a high-dimensional feature space (S. v. Wikipedia 2014). The SVM classifier constructs a hyperplane or set of hyperplanes in a high- or infinite dimensional space. A good separation is achieved if the hyperplane has a large distance to the nearest training data points of any class (functional margin), since in general the larger the margin the lower the generalization error of the classifier.

2.6.1 Linear Support vector machine

The linear support vector machine (Linear-SVM) is the simplest case of SVMs and can be used to classify linear separable data by constructing a separating hyperplane. Suppose there are labeled training data

$$\{\boldsymbol{x}_i, y_i\}, i = 1, \dots, l, y_i \in \{-1,1\}, \boldsymbol{x}_i \in \boldsymbol{R}^d \tag{21}$$

and a hyperplane which separates the positive and negative data. The points $\boldsymbol{x}$ which lies on the hyperplane satisfy $\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$, where $\boldsymbol{w}$ is the normal of the hyperplane and $|\boldsymbol{b}|/||\mathbf{w}||$ is the perpendicular distance from the hyperplane to the origen, and $||\boldsymbol{w}||$ is the Eucledian norm of $\boldsymbol{w}$. For the linear separable case, the goal of the algorithm is to find the separating hyperplane with the largest margin. This can be formulated as follows:

$$\boldsymbol{x}_i \cdot \boldsymbol{w} + b \geq +1 \; for \; y_i = +1 \tag{22}$$

$$\boldsymbol{x}_i \cdot \boldsymbol{w} + b \leq -1 \; for \; y_i = -1 \tag{23}$$

These can be combined into one set of inequalities:

$$y_i(\boldsymbol{x}_i \cdot \boldsymbol{w} + b) - 1 \geq 0 \; \forall i \tag{24}$$

The points for which the equality (41) holds are placed on the hyperplane $H_1: \boldsymbol{x}_i \cdot \boldsymbol{w} + b = 1$ and the point for which the equality (50) holds are placed on the hyperplane $H_2: \boldsymbol{x}_i \cdot \boldsymbol{w} + b = -1$, they are called support vectors. The distance of the hyperlane H_1 and H_2 from the separation hyperplane is $d_+ = d_- = 1/||\boldsymbol{w}||$ and the margin is $2/||\boldsymbol{w}||$. To maximize the margin, $||\boldsymbol{w}||$ has to be minimized subject to the constraints (24). This problem can be reformulated by introducing Lagrange multipliers $\boldsymbol{\alpha}$ to the Lagrangian:

$$L_P = \frac{1}{2}\left|\left|\boldsymbol{w}\right|\right|^2 - \sum_{i=1}^{l} \alpha_i y_i (\boldsymbol{x}_i \cdot \boldsymbol{w} + b) + \sum_{i=1}^{l} \alpha_i \tag{25}$$

The Lagrangian L_p has to be minimized with respect to $\boldsymbol{w}$, b, and simultaneously require that the derivatives of L_p with respect to all the α_i vanish, all subject to the constrains $\alpha_i \geq 0$. Now it is a quadratic programming problem which can be solved by standard quadratic programming techniques and programs. The solution can be read in (Burges 1997). The vector $\boldsymbol{w}$ can be expressed as a linear combination of the training vectors:

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x_i} \tag{26}$$

The problem can be reformulate in the “dual” problem which maximizes L_p subject to the constrain that the gradient of L_p with respect to $\boldsymbol{w}$ and b vanish, and the subject also to the

constrain that the $\alpha_i \geq 0$. Requiring that the gradient of L_p with respect to $\boldsymbol{w}$ and $\boldsymbol{b}$ vanish give the condition:

$$\sum_{\boldsymbol{i}} \alpha_i y_i = 0, 0 \leq \alpha_i \leq C \tag{27}$$

This can be substituted in (52) to give

$$L_D = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \boldsymbol{x_i} \cdot \boldsymbol{x_j} \tag{28}$$

(Burges 1997)

2.6.2 RBF Support vector machine

The linear-SVM algorithm can be extended by using non-linear functions as hyperplanes. This is done with the so called kernel-trick. The dot product $\boldsymbol{x_i} \cdot \boldsymbol{x_j}$ is replaced by a nonlinear kernel function $k(\boldsymbol{x_i}, \boldsymbol{x_j})$**.** The hyperplane can now separate the positive and negative samples in a higher feature space. A common used nonlinear kernel is the Gaussian radial basis function (RBF) kernel:

$$k(\boldsymbol{x_i}, \boldsymbol{x_j}) = exp\left(-\boldsymbol{\gamma} \left|\left|\boldsymbol{x_i} - \boldsymbol{x_j}\right|\right|^2\right), for\ \gamma > 0 \tag{29}$$

$$\gamma = \frac{1}{2\sigma^2} \tag{30}$$

The fact is caused by the complexity of the RBK kernel which is lower than for polynomial kernels (C. Hsu, C.Chang, C. Lin 2003).

2.6.3 Grid search method for parameter selection

One of the most important parts of support vector machines (SVM) modeling is the parameter selection. In this approach the grid search method is used to estimate the optimal parameter

which maximizes the classification accuracy. By selecting a RBF kernel function, the regularization constant C and the kernel hyperparameter γ have to be determined.

The grid search method is taking m values in C and n values in γ to form a $m\ x\ n$ grid. The values are used to estimate the performance of trained SVMs in a cross-validation model. The optimal parameter combination is chosen depending on the maximum performance.

- Continue
- Computation cost, …

(C. Qubo 2014)

2.7 Optical character recognition

2.8 Recognition of electronic components

dsf

2.9 Life Cycle Inventory (LCI) analysis

Sd

2.9.1 Categorization of WEEE and recycling potential of PCB waste

ert

2.9.2 Recycling potential of electronic parts from PCB waste

ret

2.9.3 Reuse potential of electronic parts from PCB waste

2.9.4 International Reference Life cycle Data System (ILCD) format

3. Methods for electronic component recognition

iop

3.1 Image preprocessing

S

3.1.1 Image rotation correction

To bypass the restriction of rotation invariant features for object recognition, the rotation angle of the printed circuit board images were determined. Since there is no fixed printed circuit board orientation, the orientation is set by invariants of 90 degree whereas most of the electronic parts are horizontal or vertical alligned. The whole process is based on the assumption that Conducter tracks and electronic parts are mostly horizontal or vertical aligned and there structure and borders producing more horizontal and vertical edges than edges with different orientations. The rotation angle estimation is based on the rotation property of a discrete Fourier transform. The DFT of an image rotated by and angle Θ is the DFT of the unrotated image, rotated by the same angle Θ. The rotation property of a DFT is derived in (Maria Petrou, Costas Petrou 2010) and therefore omitted here. The image rotation correction process is shown in Figure 3.

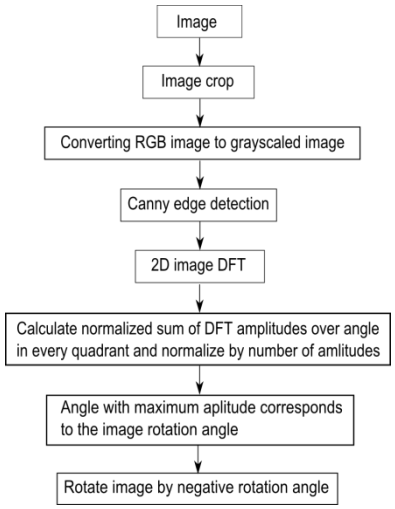


Figure 3: Image rotation correction process

At first the Image is cropped to a squared image [2000 x 2000] to reduce the process runtime. The RGB image is converted to grayscaled image and canny edge detection is applied. Afterward a 2D DFT is computed from the edge image. To estimate the rotation angle, the amplitude of the shifted 2D FFT image is summed up over discretized angles and normalized by number of amplitudes per angle step. The discretization is done in steps of 0.25 degree from 0 to 360 degree which results in a discretization error of 0.125 degree. The maximum of the normalized sum of amplitudes over the angle corresponds to the image rotation angle. With this process the rotation angle can be estimated with invariants of 90 degree image rotation. An example of a rotated image by 3 degree, the edge image and amplitude of discrete Fourier transform is shown in Figure 4, Figure 5 and Figure 6. The accuracy of the angle estimation was not investigated in detail but inaccuracy could not be determined by eye.

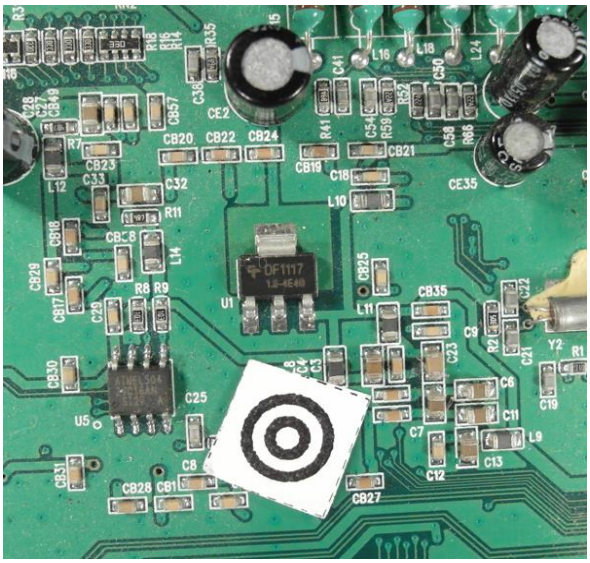


Figure 4: Image rotated by 3.0 degree

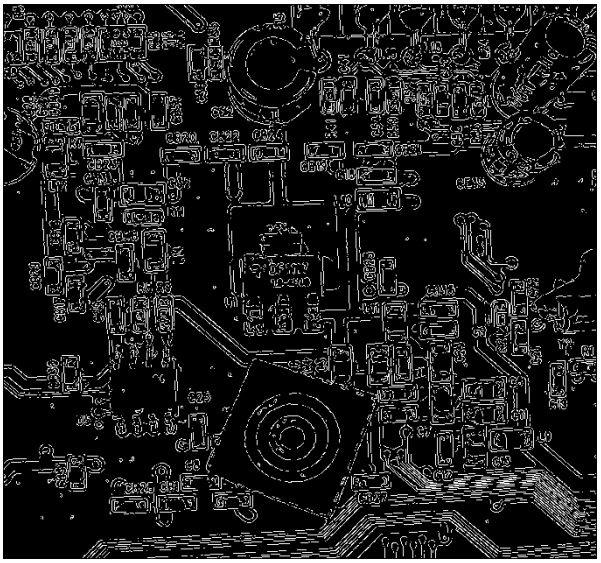


Figure 5: Canny edge image of the rotated image

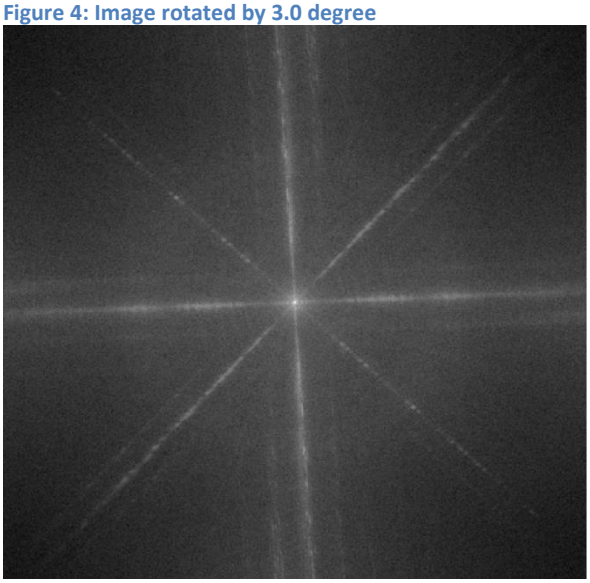


Figure 6: Shifted DFT of the rotated image (logarithmic representation)

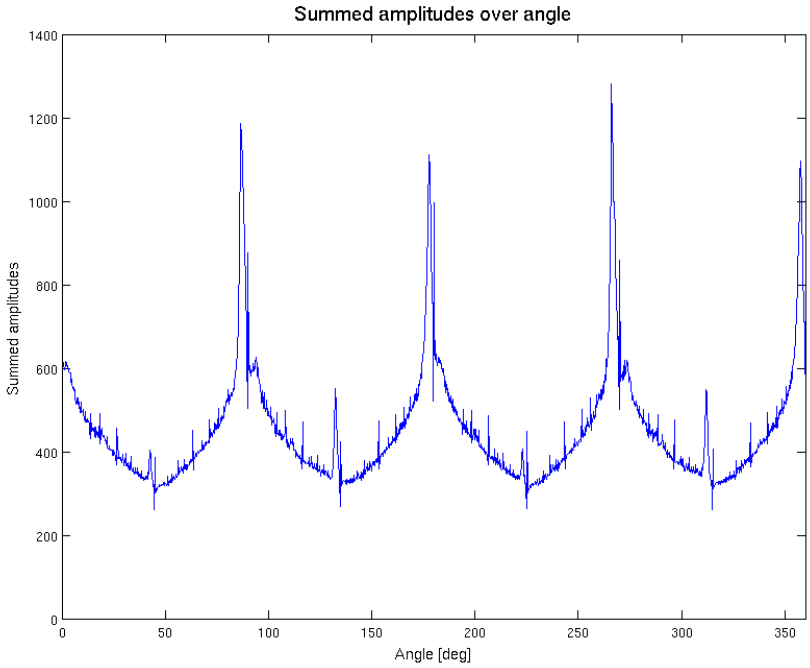


Figure 7: Summed amplitude over angle (invariants by 90 degree)

- Linien werdan auf punkte abgebildet

3.1.2 Scaling determination based on scaling symbol

To bypass the restriction of scale invariant features for object recognition, the scaling of the printed circuit board images were determined using a scaling symbol.

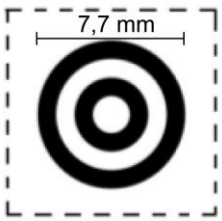


Figure 8: Scale symbol

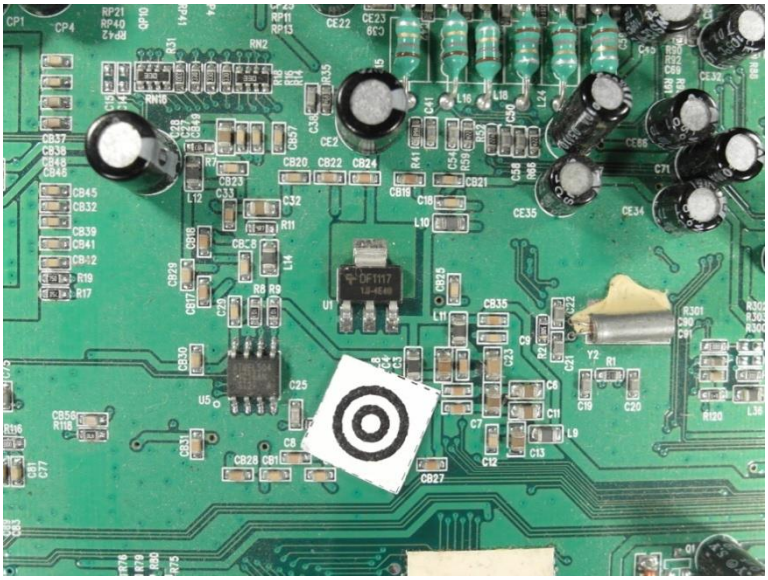


Figure 9: Scale symbol placed on the board

The scaling symbol is shown in Figure 8. The whole scaling determination process is shown in Figure 10.

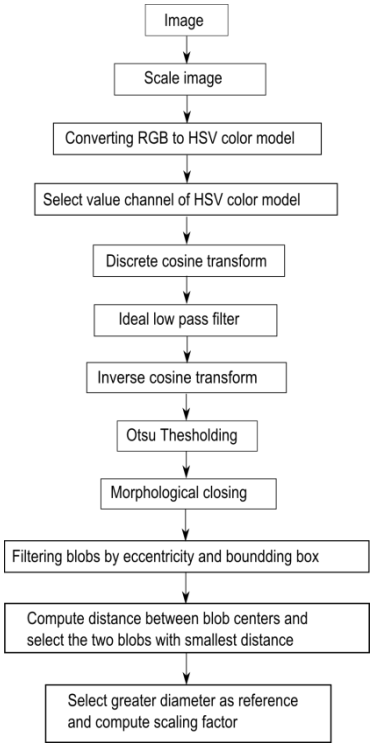


Figure 10: Scaling determination process

At first the image is converted from the RGB color model to the HSV color model and the brightness channel (value channel) is used to make a discrete cosine transform. The discrete cosine transform is frequently used in image compression such as the JPEG format. The discrete cosine transform is similar to the discrete Fourier transform but uses only cosine functions as kernels. The discrete cosine transform is shown in Equation (31) and (32) (Rafael C.Gonzalez 2008).

$$T(u,v) = \sum_{x=0}^{n-1}\sum_{y=0}^{n-1} g(x,y)\alpha(u)\alpha(v)\cos\left[\frac{(2x+1)u\pi}{2n}\right]\cos\left[\frac{(2y+1)v\pi}{2n}\right] \tag{31}$$

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{n}} & for\ u = 0 \\ \sqrt{\frac{2}{n}} & for\ u = 1, 2, \dots, n-1 \end{cases} \tag{32}$$

$$\alpha(v) = \begin{cases} \sqrt{\frac{1}{n}} & for\ \ v = 0 \\ \sqrt{\frac{2}{n}} & for\ \ v = 1, 2, \dots, n-1 \end{cases} \tag{33}$$

To suppress illumination changes, an ideal low pass filter is applied in the frequency domain in which the first 10 x 10 cosine coefficients were discarded. Afterwards the inverse cosine transform is applied to get the image in time-domain. To extract the two dark circles of the scaling symbol, Otsu's method is used to automatically perform thresholding. To avoid salt and pepper noise, a morphological closing operator (5x5) is applied. The image is inverted and the eccentricity and bounding boxes are determined of the blobs. All blobs inside the eccentricity interval and inside the diameter interval are maintained, all others are discarded.

$$Maintained\ blobs = \{blobs,\ eccentricity_{min} < eccentricity \wedge diameter_{min} < diameter < diameter_{max}\} \tag{34}$$

- eccentrcity min angeben

To find the center of the scaling symbol, the distances between the centers of all blobs are calculated and the two blobs with the smallest distance are the inner and outer dark rings of the scaling symbol. The outer diameter of the larger blob is used as reference to calculate the image scale.

$$imagescale = \frac{diameter\ [pixel]}{diameter\ [mm]} \tag{35}$$

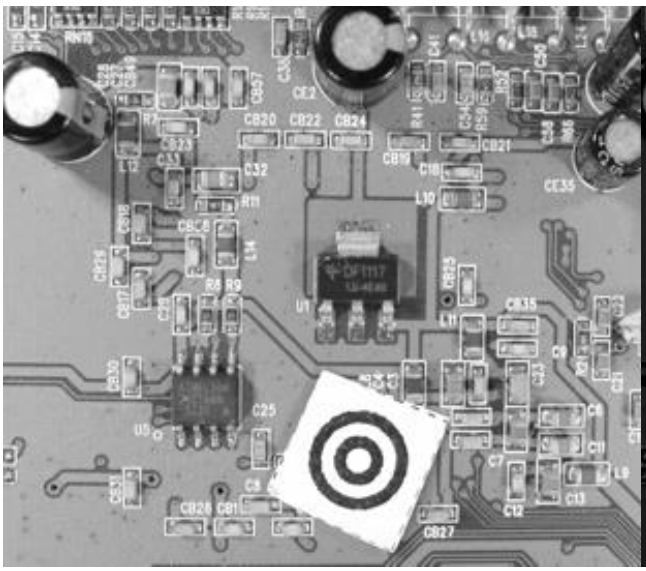


Figure 11: Value channel (brightness) of HSV color image

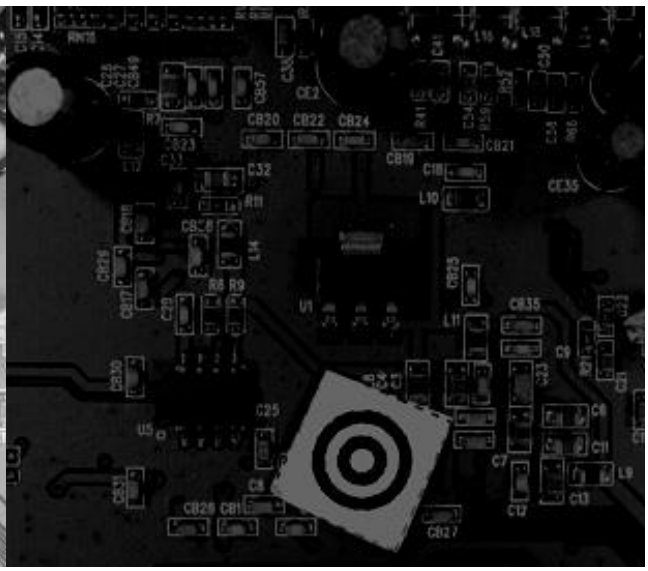


Figure 12: Cosine transform filtered image

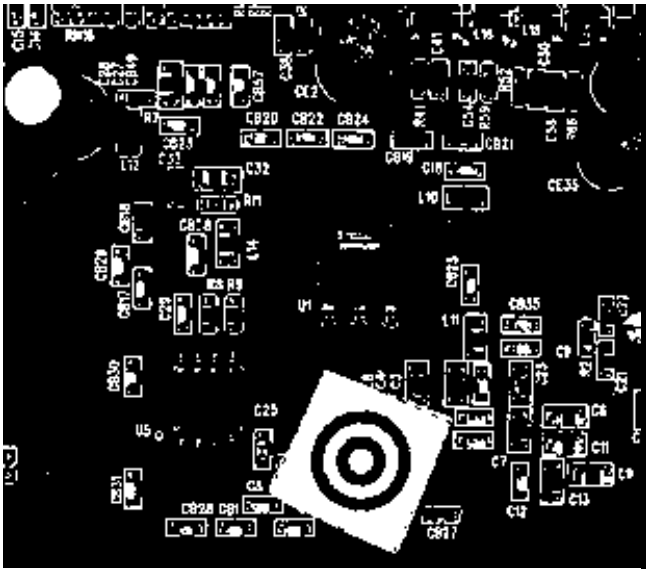


Figure 13: Otsu thresholding

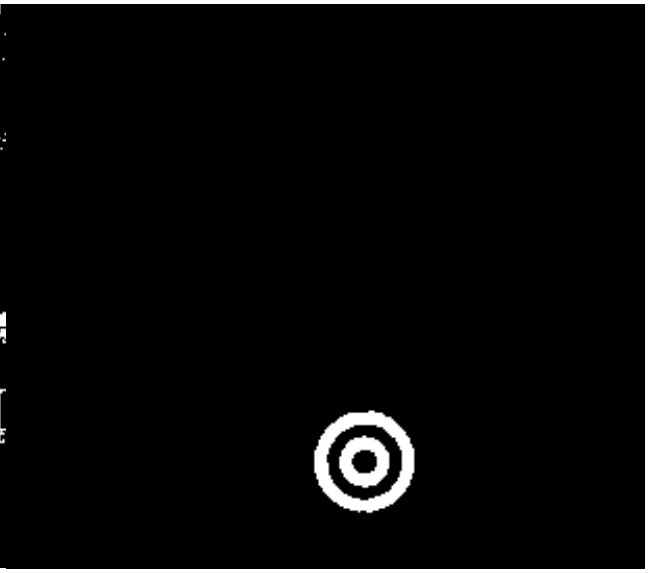


Figure 14: Blobs of the scaling symbol

3.1.3 Image resolution for feature extraction

The resulting features quality of feature extraction algorithms depend on the resolutions of the images. In general higher image resolutions improve the feature precision but also increase the run time and require more memory. Therefore a trade off between a high image resolution on one hand and memory usage and runtime on the other side must be found. In this approach the image resolution depends on the size of the component. Smaller components require a higher resolution than larger ones because there images contain more details.

- entropy

In this approach the resolution depends on the components area and the feature extraction algorithm.

$$area_{component}\,[mm^2] = width_{component}[mm] * hight_{component}[mm] \qquad (36)$$

$$PPMM(area_{component}) = a * \exp(-b\,(area_{component}\,[mm^2]) - c)\,[ppmm] \qquad (37)$$

The algorithm dependent resolution parameters are defined in Table 1.

Table 1: Feature extraction algorithm based resolution parameter

	a	b	c
Fourier coefficients based feature extraction	5	0.003	15
Histogram based feature extraction	10	0.003	10
Segment based feature extraction	19	0.005	1
PCA reconstruction	18	0.005	2

The area and algorithm dependent resolution is plotted in Figure 15.

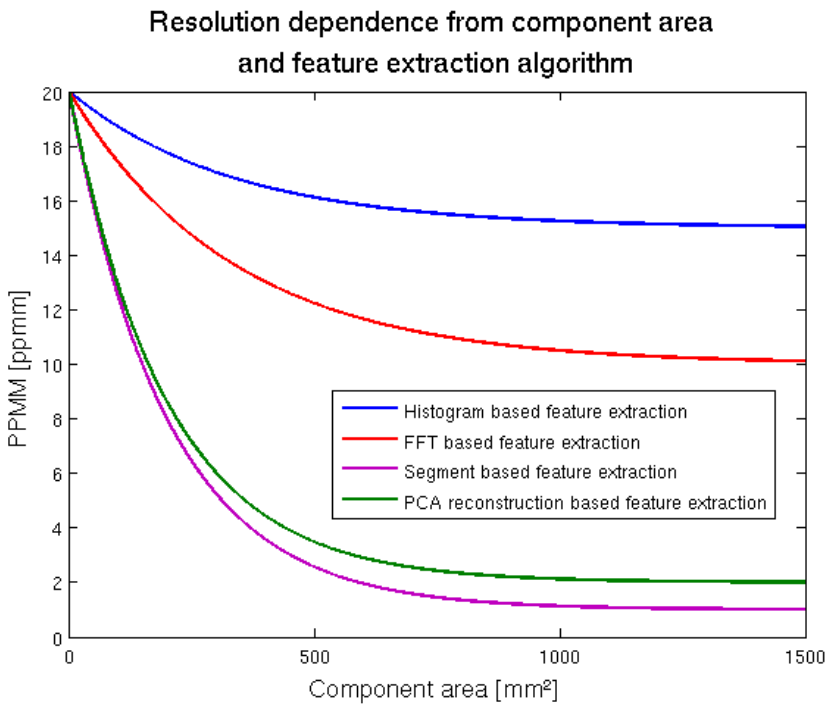


Figure 15: Image resolution

- Abbildung korrigieren

3.2 Electronic component detection

- Laufzeit, suchraum eingrenzen

Asdsad

3.2.1 PCB board segmentation

One of the steps before detecting electronic components is the segmentation of the PCB board to reduce search area for electronic components. In this approach the images of the PCB board where acquired where the PCB boards where placed on a white sheet what results in a white/bright background. In this approach the process flow shown in Figure 16 is applied.

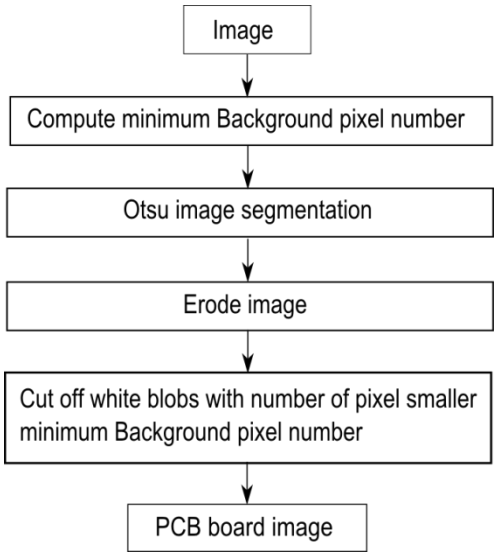


Figure 16: PCB board segmentation process flow

At first the minimum of background pixel is defined by 5% of the image pixel to do not cut of white regions from the PCB board.

$$\#Backgroundpixel_{min} = 0.05 * \#Imagepixel \quad (38)$$

Than Otsu segmentation is applied, followed by a morphological erode step with a 10x10 kernel to separate white regions from the PCB board which are connected with the background. In the last step all blobs with the number of pixels greater than the minimum background pixel number $\#Backroundpixel_{min}$ are cut off whereby all remaining regions are mainly PCB regions.

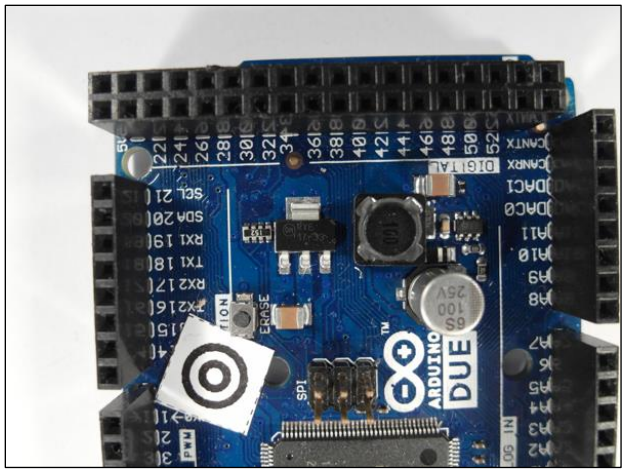


Figure 17: Acquired PCB image

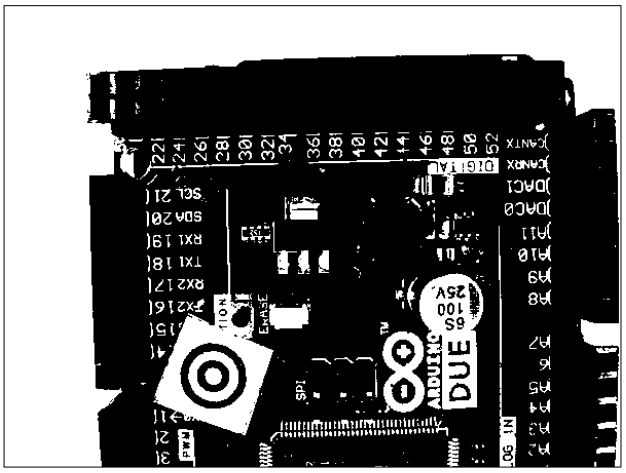


Figure 18: Otsu segmentation

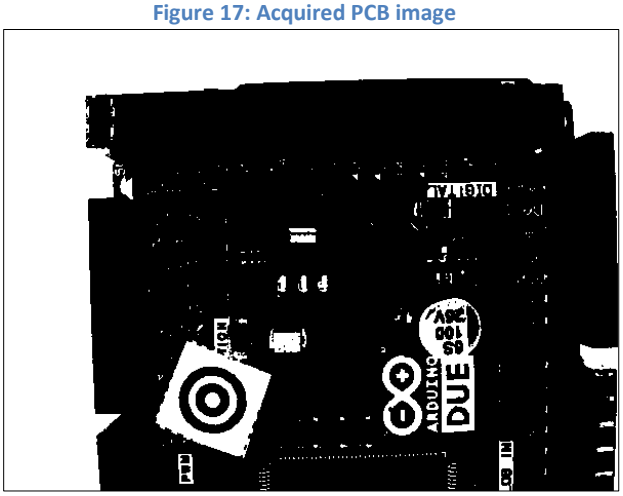


Figure 19: Morphological eroded image with 10x10 kernel

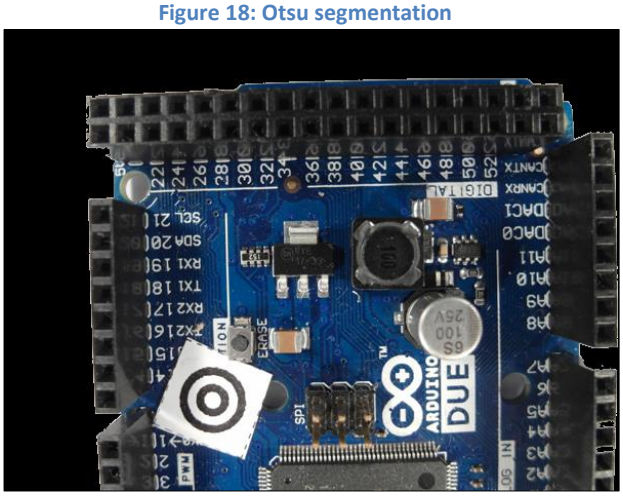


Figure 20: Segmented PCB board image

3.2.2 Color based PCB surface detection

To classify an electronic component it is necessary to know the position of the electronic component on the PCB board. One possibility process step is the segmentation of the PCB surface based on the color and distribution of the surface pixel over the PCB image.

This approach is based on the following assumption of PCB surfaces:

- Most PCB surfaces have striking colors compared to the color of the electronic components or PCB markings. That results in the mostly colored isolating protection lacquer whereas often used colors are green, blue, orange, red, etc.

- The number of surface pixel cluster is high compared to other pixel clusters cased on the mostly large surface area compared to individual components
- The surface pixels form mostly large areas of the PCB surface what results in a small number of segment blobs compared to other clusters
- The surface segments form mostly contiguous areas with the result that the number of edge pixels is smaller compared to other segment clusters

The process flow is shown in Figure 21.

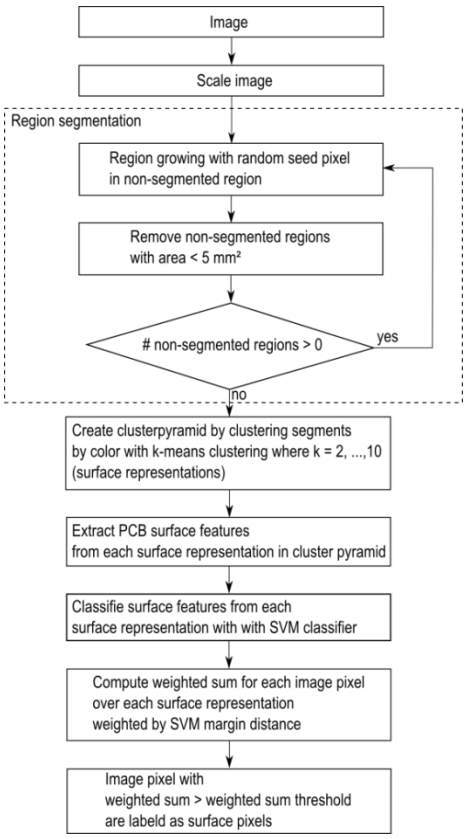


Figure 21: PCB surface segmentation process flow

The image is scaled to a resolution of $5\,pixel/mm$ to speed up the PCB surface detection process. In the second process step a region growing approach is used to divide the image in regions with similar color. The seed points of the region growing algorithm are chosen

randomly under the requirement that the seed points are placed in the non-segmented image region. The criterion to stop the growing process of a seed point is the similarity threshold value which is the Euclidian distance between the color of the neighboring pixel and the average color of the region. Exceeds the distance a distance threshold value of 0.2 the neighboring pixel will not be considered as a region pixel. The growing process of a seed point stops if no neighboring is considered to the region. The region growing process is explained in detail in chapter 2.2.2. After segmenting a region, all non-segmented regions with an area smaller than $5mm^2$ are removed from the non-segmented region to speed up the process. If there are still non-segmented regions, the region growing process is repeated with a new randomly selected seed point in the non-segmented region. If all image regions are segmented or rejected from the non-segmented region caused to their small region area the process stops. The first 200 segments during a region segmentation process is shown in .

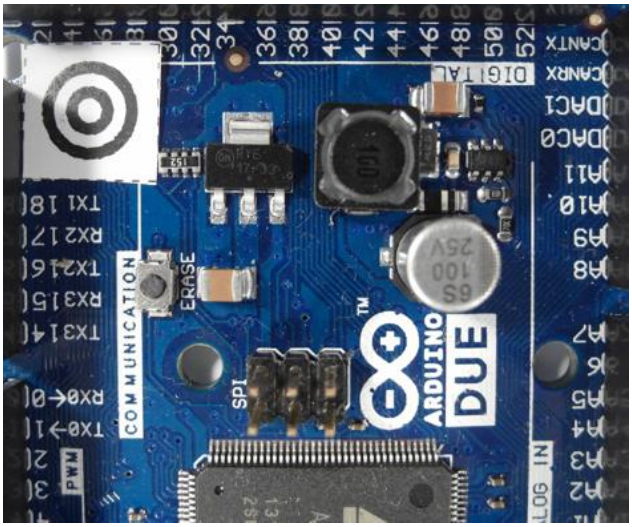


Figure 22: Original image

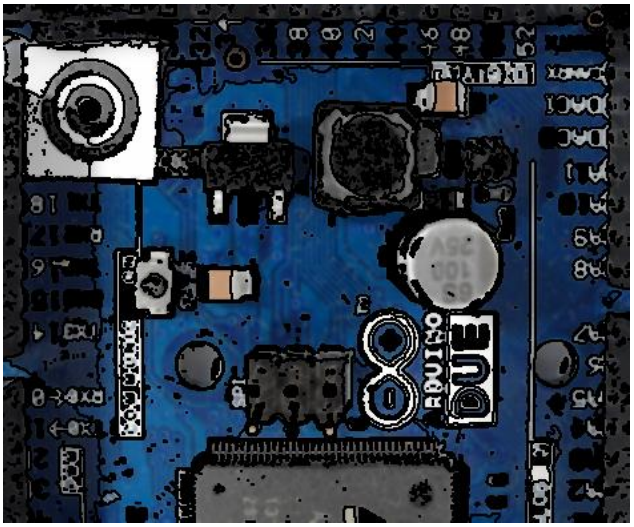


Figure 23: First 200 image segments based on region growing approach

After region growing, the segments are clustered based on their color. A cluster pyramid is drawn up where the number of clusters increases by one on each level of the cluster pyramid. The k-means clustering algorithm is used with randomly selected initial set of k means. The k-means clustering algorithm is explained in detail in chapter 2.2.3. The

maximum number of cluster levels of the pyramid is set to ten ($k_{max} = 10$) and shown in Figure 24.

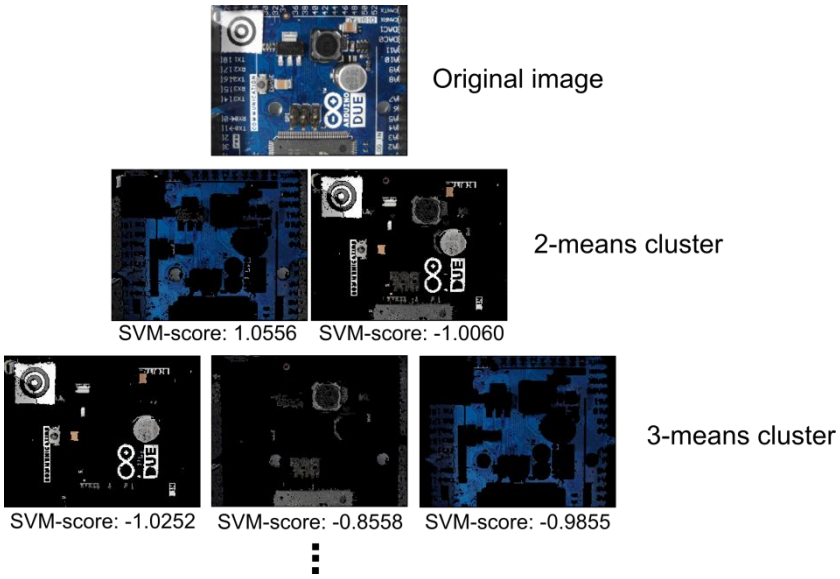


Figure 24: PCB surface cluster pyramid

After creating the cluster pyramid for all 54 surface representations ($2 + 3 + 4 + \cdots + 10 = 54$) the surface features are extracted. The nine surface features are:

- Color #3
- Number of surface representation pixel normalized by the maximum number of surface representation pixel from the same pyramid level (# 1)
- Number of surface representation edge pixels normalized by the maximum number of surface representation edge pixels from the same pyramid level (Number of edge pixel is determined based on first derivative kernel in the gray scaled image) (# 1)
- Number of blobs in the surface representation normalized by the maximum number of blobs in a surface representation from the same pyramid level (# 1)
- Elements of the covariance matrix of color pixels from surface representation (# 6)

To separate good surface representations from bad ones, each surface representation is classified according to the nine features with an RBF-Kernel SVM ($\sigma = 1.2$).

To train the RBF-Kernel SVM, each surface representation in the cluster pyramid of 102 images was labeled according to their goodness of PCB surface representation. Surface representations in the cluster pyramid were labeled with 1 if the pixels represent mainly the surface and -1 if the pixels in the cluster are mainly pixel from electronic components or PCB markings. That results in a set of $54 * 102 = 5508$ clusters whereas 870 clusters where labeled as PCB surface and 4638 clusters where labeled as non-PCB surface representations. Ambiguous cluster representations were labeled as non-PCB surfaces.

The distances of the feature vectors from the decision boundary of the RBF-Kernel SVM where treaded as scores $s_i, i = 1, \dots, 54$ whereas a high positive score identifies good surface representations and low negative scores represent bad surface representations. For each pixel of the image, the sum of scores over all 10 levels is computed. The scores are treaded as weights of the surface representation in which the pixel was included. If pixels are not included in a cluster of a pyramid level because the region in which the pixel was included, was rejected caused by the small region area, the score is set to zero. Each Pixel $f(x, y)$ at the position x, y with score sum $w(x, y)$ greater than the weighted sum threshold w_{thr} is selected in the PCB surface set S.

$$w(x, y) = \sum_{i=1}^{54} s_i(x, y) \tag{39}$$

$$S = \{f(x, y) \mid w(x, y) > w_{thr}\} \tag{40}$$

In this approach w_{thr} was set to zero. All selected PCB surface pixel form the PCB surface. The weighted sum of scores of the image Figure 25 is shown in Figure 26. It is clearly seen that the PCB surface pixel have higher score values than others.

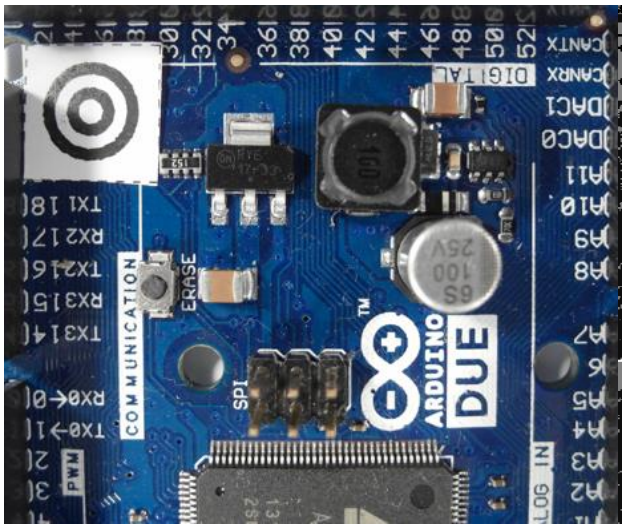


Figure 25: original PCB image

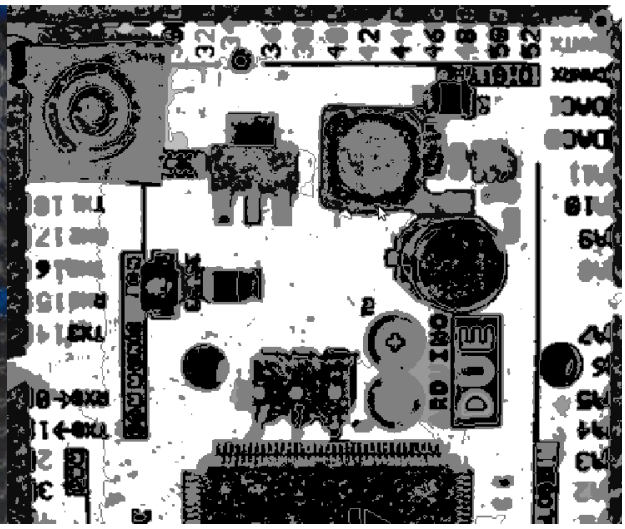


Figure 26: Sum of RBF-kernel SVM scores $w(x, y)$ (grayvalues are scaled between -20 and 20)

3.2.3 Electronic component detection based on normalized 2-D cross-correlation

Template matching is a technique in digital image processing for finding regions in an image that match a smaller image template. The normalized cross correlation is fast way of matching templates in an image and is used in many object detection approaches. A detailed description about pattern matching with normalized 2-D cross correlation is done in chapter 2.2.4.

In this approach the templates where generated by the train images of the electronic components. For each component the average values over all training images in all three color channels where computed. The average image is computed in the HSV color space and treaded as the component template. The template of the DIP14 component is shown in Figure 17.

Figure 27: Image template for DIP14 component (RGB color space)

In determining the image resolution a trade of between the computation time and spatial image resolution has to be made. In this approach the spatial resolution depends on the

component surface area. The relation between spatial image resolution and component surface for the normalized 2D-cross correlation is shown in Figure 28.

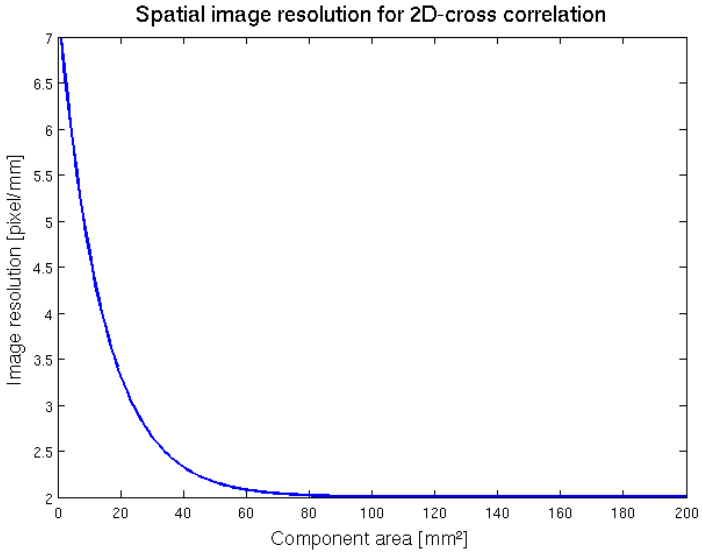


Figure 28: Spatial image resolution for 2D-cross correlation

- Plot: area=surface area

To perform the 2D cross-correlation the examined image is also converted to the HSV color space and the cross-correlation is performed in all color spaces. The average correlation values over all three color spaces are determined and filtered by a 2D Gaussian kernel to get a score map $p(x, y)$. The Gaussian kernel has a size of 5x5 pixel and $\sigma = 1.5$. Scores $p(x, y)$ greater than a correlation threshold $Corr_{thr}$ are treated as a set of potential component positions S. The correlation threshold $Corr_{thr} = 0.4$ seems to be a good trade of between false positive rate and true positive rate.

$$S = \{f(x, y) \mid p(x, y) > Corr_{th}\} \tag{41}$$

- Threshold bestimmen (ROC)

An image and its determined potential component positions for the DIP14 component is shown in Figure 29. It can be seen that the two DIP14 parts in the image where recognized but also a false positive component position was selected.

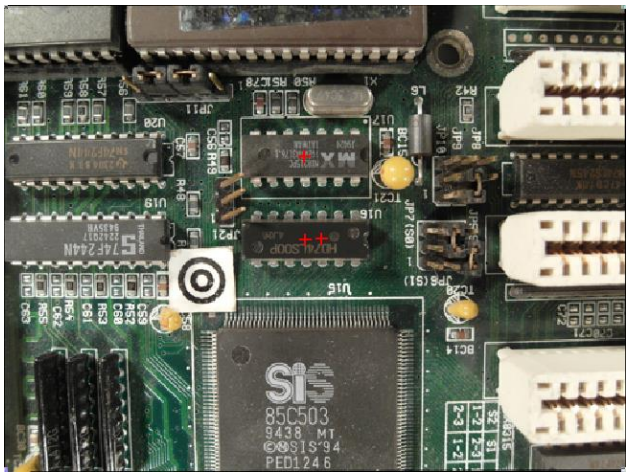


Figure 29: Determined potential component positions for DIP14 component

– change plot

3.2.4 Electronic component detection based on 3D range image

3.3 Featureextraction

3.3.1 Fourier coefficients based feature extraction

Every periodical infinite signal can be decomposed in ….

Fourier descriptors as features were used in already used in applications for face recognition and object recognition (Campos 2000).

The idea to use Fourier coefficients as features comes from the representation of solder joints by most electronic component images. Many computer vision systems for solder join detection, localization and segmentation have been develop. Specular reflections of solder joint depending on small changes in viewing direction and different shape and size of the solder joints make it difficult to create a stable recognition system (Tianshoul 2012). Many electronic components consist of several equidistant arranged solder joints. An example is the widely used DIP14 package seen in Figure 30. Since the solder joints appear in the grayscaled image as bright equidistant spots they should be representative frequencies in the 2D Fourier spectrum with the period around the solder joint distance.

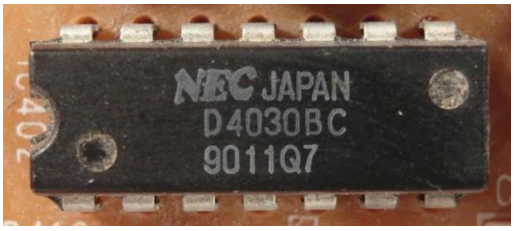


Figure 30: DIP14 package with equidistant solder joints

The 2D discrete Fourier transform for an M x N image is defined as

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})} \tag{42}$$

$u = 0, 1, 2, \dots, M-1$ and $v = 0, 1, 2, \dots, N-1$ where $f(x,y)$ is the image of size M x N (Rafael C.Gonzalez 2008). The Fourier coefficients are in general complex numbers consisting of real and imaginary part. The real part represents the cosine and the imaginary the sinus proportion of the signal. The M x N image consists of M x N Fourier coefficients which produces 2 x M x N frequency features which is a large number of features that can be used. To increase execution time of the classifier and decrease recognition rate, a subset of low frequency features is extracted. Further research shows that spatial frequencies with lower frequency represent global information about the shape such as general orientation and proportion. The visual information is represente

Since the solder joints are the main focus for frequency feature, the solder joint distance of electronic components is used as a measure of minimal frequency period. In our feature extraction all Fourier coefficient (real and imaginary part) with a frequency under the cutoff frequency are used as features.

$$f_{cutoff} = \frac{1}{T_{cutoff}} = \frac{1}{0.50\ mm} = 2\ mm^{-1} \tag{43}$$

The numbers of features depend on the size of the component image.

$$\#frequency\ features = \left\lceil \frac{length\ [mm]}{T_{cutoff}[mm]} + 1 \right\rceil * \left\lceil \frac{width\ [mm]}{T_{cutoff}[mm]} + 1 \right\rceil \tag{44}$$

Abtastthorem (+1)

Another interesting feature extraction based on wavelets could analyze frequencies and there temporal occurrence which could improve the classification results. A view on that topic was done in the prospective section **Fehler! Verweisquelle konnte nicht gefunden werden.**.

- Energie in niedrigen frequnzen -> hohe information (paper)

3.3.2 Histogram based feature extraction

Color image segmentation algorithms for automated optical inspection in electronics have already been investigated (Tarnawski 2003). Electronic components varying in color, such as several tantalum capacitors, ICs or SMD electrolyte capacitors. To find representative features the color model has to be defined. In this system, the HSV (hue-saturation-value) color model was used because the channels are not that strong correlated such as in the RGB color model and relative stable against illumination changes or shadows (H. Cheng, H. Jiang, Y. Sun, Jingli Wang 2000), (Noor. A. Ibraheem, Mokhtar M. Hasan, Refiqul Z. Khan, Pramod K. Mishra 2012). Histogram based features are features which depend on the probability distribution of the pixels over the color values. In the histogram based feature extraction 10 equidistant bins are defined in each color channel (hue-saturation-value) and the pixel distributions are determined and normalized by the number of pixels. The values correspond to the probability density

function of the gray value. All ten bin values are use as features that results in 30 color features. The histogram of a tantalum capacitor is seen in Figure 31, Figure 32, Figure 33 and Figure 34.

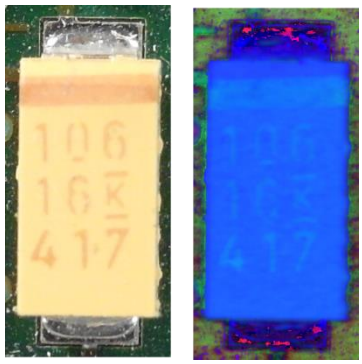


Figure 31: Tantalum capacitor in RGB color model (left) and HSV color model (right)

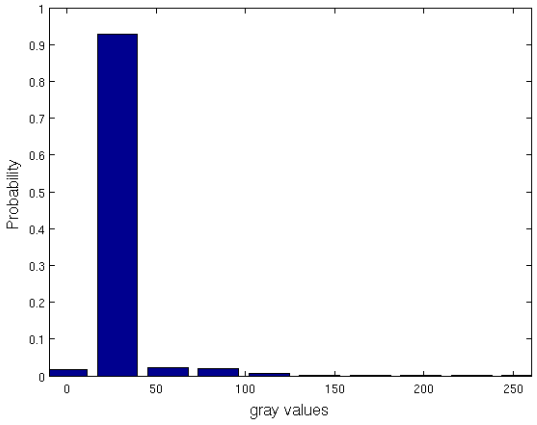


Figure 32: Normalized histogram of hue channel (tantalum capacitor)

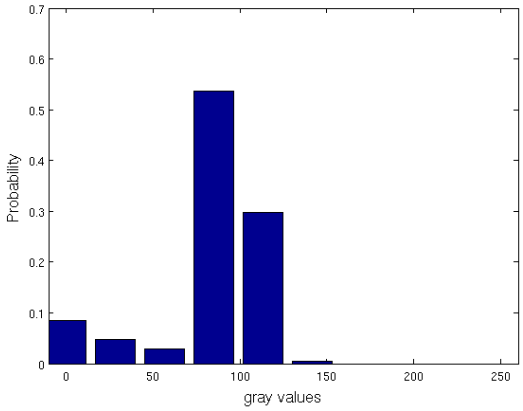


Figure 33: Normalized histogram of saturation channel (tantalum capacitor)

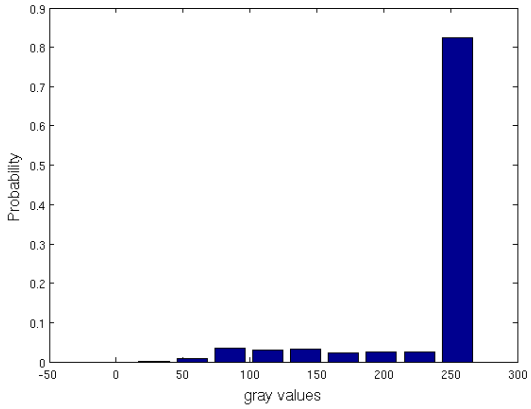


Figure 34: Normalized histogram of value channel (tantalum capacitor)

3.3.3 Segment based feature extraction

The segment based feature extraction is based on the idea that electronic components can be identifies by striking color regions. One approach to extract information about spatial proximity of pixels is the region growing algorithm. The region growing starts with seed points which pixel position is the most important drawback.

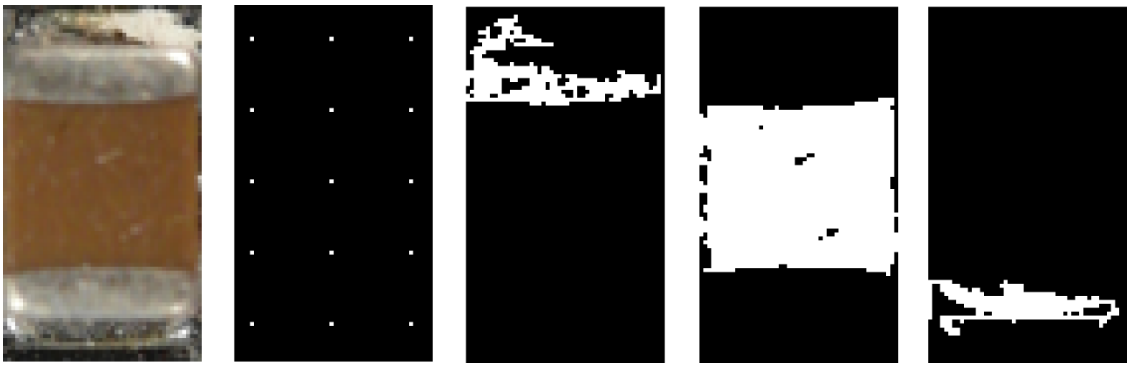
In this feature extraction algorithm, the seed points are uniformly distributed over the part image. The region growing and feature extraction of the segments is done in HSV color space. The distance between the seed points depends on the size of the component which is based on the assumption that smaller components consist of smaller color regions than big components. The equation for the distances is specified in (45).

$$A_{component} = length_{component} * \ width_{component} [mm^2] \quad (45)$$

$$\Delta_{seed} = 0.02 * A_{component} + 0.5 \ [mm] \quad (46)$$

In the region growing segmentation approach the neighboring pixel of the seed pixel are added to the segment if the distance between the color of the seed point and the neighboring pixel is smaller than a certain value. Further the neighboring pixels of the new segment are added to the segment if their distance to the color mean of the segment is smaller than a certain value. This process is iterated until no more pixels are added to the new segment (Maria Petrou, Costas Petrou 2010).

One example is the Multi-layer ceramic capacitor (MLCC) shown in .



- Abbildung korrigieren

Seven Features are extracted for every segmented region which are the x-coordinate of center of gravity, y-coordinate of center of gravity, bounding box height, bounding box width and the arithmetic mean color value in all three color channels.

- Border to seeds in picture
- Formeln region growing

3.3.4 PCA reconstruction error based feature extraction

Object detection based on image reconstruction with Principal Component Analyses was already applied for pedestrian recognition. (L. Malagón-Borja, Olac Fuentes 2007). A similar approach was used to extract a PCA reconstruction feature. In that system the PCA reconstruction is based on edge images of the parts. At first a subset of the training images of parts are used to find principal components which can compress optimally only the kind of images that were used to compute the principal components.

- Übergang prüfen!!!

A set of PCs from a set of images from one component reconstruct the images of the same component better than other types of images. The fact can be observed in the images in Figure 35 and can be used to create a feature which represents the difference between the reconstruction error of the projection into the component PCs and the non-component PCs.

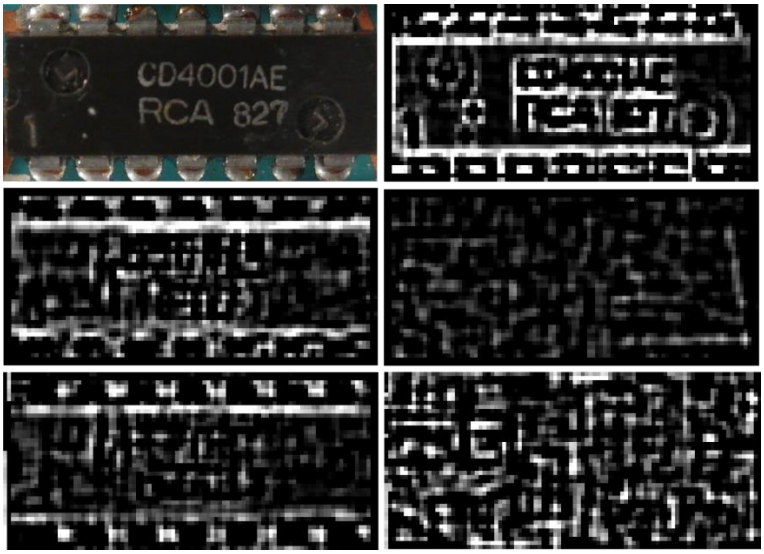


Figure 35: DIP14 (top, left), DIP14 edge image (top, right), DIP14 reconstruction with component PCs (middle, left), DIP14 reconstruction with non-component PCs (middle, right), unit matrix projection into component PCs (bottom, left), unit matrix projection into non-component PCs (bottom, right)

In this approach the component images and non-component images are scaled according to the size of the component. After the RGB images are converted to grayscaled images and the image intensity values are adjusted for contrast improvement. To obtain a feature that contains information about the edges the edge image was created by applying a Laplacian of Gaussian (LoG) filter. The projection matrix $\boldsymbol{P}_{ep}$ and the mean $\boldsymbol{\mu}_{ep}$ are computed from a subset of component images and the projection matrix $\boldsymbol{P}_{en}$ and mean $\boldsymbol{\mu}_{en}$ for the non-component images are computed. The reconstruction based on the component PC projection is computed by (47) and the reconstruction based on the non-component PC projection is computed by (48).

$$\boldsymbol{r}_{ep} = \boldsymbol{P}_{ep}^{T}\boldsymbol{P}_{ep}\left(\boldsymbol{e} - \boldsymbol{\mu}_{ep}\right) + \boldsymbol{\mu}_{ep} \tag{47}$$

$$\boldsymbol{r}_{en} = \boldsymbol{P}_{en}^{T}\boldsymbol{P}_{en}(\boldsymbol{e} - \boldsymbol{\mu}_{en}) + \boldsymbol{\mu}_{en} \tag{48}$$

The reconstruction error of component images projected by component PCs should be smaller for component images than non-component images. The features is the difference between the

reconstruction error projected in the component PCs and the error projected in the non-component PCs shown in (49).

$$\boldsymbol{f}_{pca} = \sum |\boldsymbol{r}_{ep} - \boldsymbol{\mu}_{ep}| - \sum |\boldsymbol{r}_{en} - \boldsymbol{\mu}_{en}| \tag{49}$$

The process is shown in Figure 36.

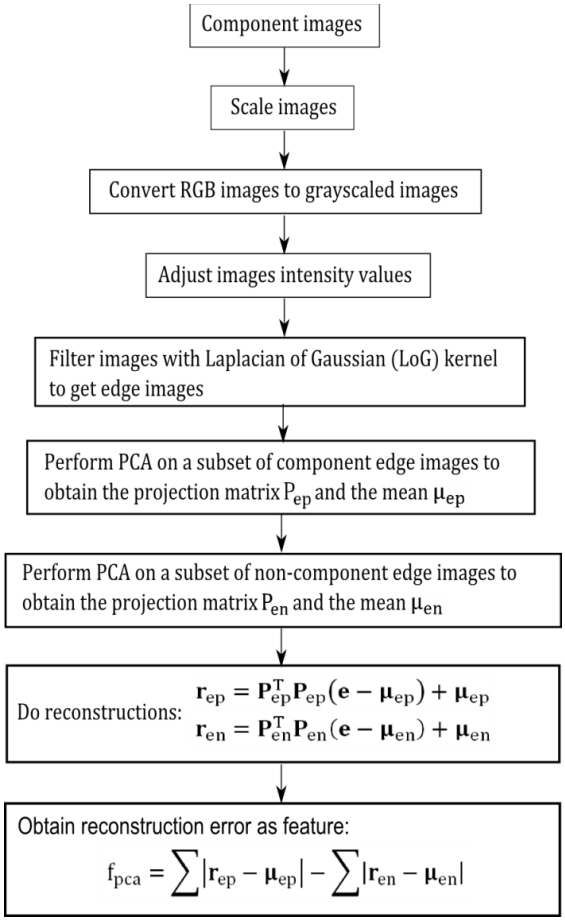


Figure 36: PCA feature construction process

Asdf

3.4 Feature selection based on Fisher score and Random forest

In practice random forest cannot handle a lot of features because it requires a lot of time to estimate the trees of the random forest and the accuracy decreases with a large number of features (Y. Chen, C.Lin 2003). This approach does feature selection in two steps. First the Fisher score is used to select a subset of features from the feature set with a large number of features. The features are selected by a fisher score threshold of 0.01. All features with a fisher score larger than the threshold value are selected for the second step. In the second step the random forest based feature selection from 2.3.2 is applied to select the most important features from step one. The process chain of the feature selection approach is shown in Figure 37.

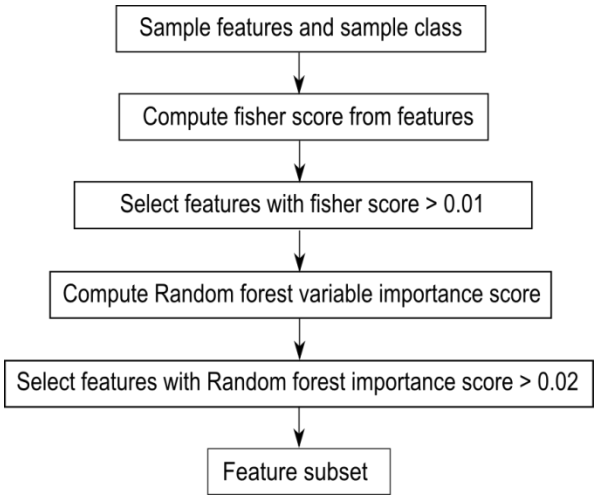


Figure 37: Feature selection process chain

3.5 Classification

3.6 Data fusion model

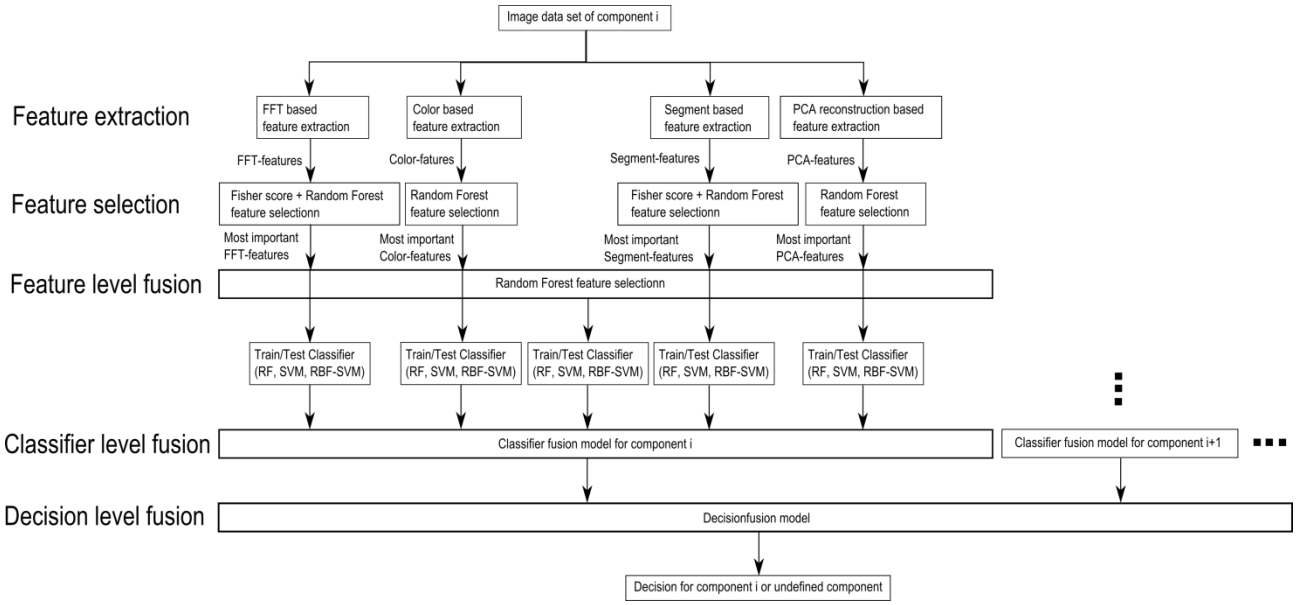


Figure 38: Data fusion model

3.6.1 Feature level fusion

The inputs for the feature selection process are the extracted features in the feature extraction process. The features are extracted from four different ranges of properties which are based on FFT features, color based features, segment based feature extraction and PCA reconstruction error based features. The feature level fusion is based on the feature selection approach whereas all the most important features of the feature selection algorithms are used as input features for a classifier in the classifier fusion step. This approach is based on the idea that a combination of features from different feature ranges can improve the estimation accuracy of a classifier.

The feature selection of the most important features from every feature range is based on the fisher score and random forest feature selection algorithm explained in chapter 3.4. The most

important features from all four features ranges are merged together and a random forest feature selection is applied to get the most important features.

One of the main problems in merging features from different feature ranges is the problem of missing values. In this approach the features based on the PCA reconstruction error and the segmentation based features contain missing vales caused by the fact that a subset of the samples is used to generate a priori knowledge for the feature extraction process. The a priori knowledge generation is explained in chapter 2.2.1. The missing value of a sample from a variable m is replaced by the median over all samples from the variable m. The replacement values are called fills (Breiman, www.stat.berkeley.edu 2014). All missing vales were used for training the classifier, so that the test data do not contain replaced values. This median replacement approach and alternatives for the replacement of missing values are explained in chapter 2.3.2.

After replacing missing values, the features with a random forest importance score greater than an importance score threshold of 0.02 are selected. The process is shown in the data fusion process chain in Figure 38.

3.6.2 Classifier level fusion

The data fusion on classifier level (classifier level fusion) is performed to make the performance more robust against the difficulties that each individual classifier may have. Combining classifiers is one of the most widely explored methods in pattern recognition and it has been shown that these techniques can reduce error rate in classification tasks (Moreno-Seco n.d.). I this approach each classifier is responsible for a specific feature subset. The first classifier rates the sample data based on the most important FFT-features, the second on the most important color features, the third on the most important segment features and the fourth on the most important PCA features. The fifth classifier rates the sample data based on the most important features of all important features of all feature extraction algorithms. The largest groups of classifier fusion methods operate on classifiers which produce so-called soft outputs. The outputs are real values in range [0, 1] (D. Ruta, B. Gabrys 2000). The random forest classifier

outputs the number of votes for a class based on the number of trees. The number of votes can be normalized by the number of trees to get a soft output.

In this approach the simple weighted vote scheme (SWV) is used to combine the five classifiers (Moreno-Seco n.d.). The soft outputs of all five classifiers are weighted by their estimation accuracy of the test samples. The output of the classifier fusion process is the soft-output P_i which represents the probability that the sample is from class i. $P_{i,k}$ represents the probability of classifier k to be component i. $P_{i,k,test}$ represents the probability of classifier k to be component i based on the true positive rate of the test set.

$$P_i = \sum_{k=1}^{5} w_{i,k} * P_{i,k} \tag{50}$$

$$w_{i,k} = \frac{\sum_{j=1}^{5} P_{i,j,test}}{P_{i,k,test}} \tag{51}$$

3.6.3 Decision level fusion

Decision-level fusion consists of merging information at higher level of abstraction. The fusion sstep combines multiple algorithms to yield a final fused decision. In this approach the outputs of the classifier fusion models at the classifier fusion level are soft outputs between 0 and 1. The output of the classifier fusion model for component i can be interpreted as the probability that the detected part is from component i. The outputs from all classifier fusion outputs are combined to make a final decision to which component the examined part belongs (Dong 2009).

There exists variety of Decision fusion techniques based on Bayesian decision theory, Dempster- shafer fusion methods, artificial neural networks or Principal component analyses.

- Explane what are they and how are they used

In this approach the Bayesian decision theory is used to combine the output of the classifier fusion level to make a final decision. The Bayes decision theory is a fundamental statistical approach to the Problem of Pattern classification. It is based on quantifying the tradeoffs between various classification decisions and the cost that accompany such decision (Duda 2001). In this approach the soft-outputs of the Classifier fusion model for each component i are measurements that are continuous random variables between 0 and 1. Under the assumption that n components exist in the recognition database with their associated n classification outputs on classifier fusion level a feature vector $\boldsymbol{x}$ can be defined in a d-dimensional Euclidian space $\boldsymbol{R}^d$ called the feature space. Let $\{w_1, w_2, \dots, w_c\}$ be the finite set of c states. In this approach every component corresponds to one state whereas one state for components that are not in the recognition database is defined that the number of states is $c = d + 1$. Let $p(\boldsymbol{x}|w_j)$ be the state-conditional probability density function for $\boldsymbol{x}$, with the probability density function for $\boldsymbol{x}$ conditioned on w_j being the true state of nature. $P(w_j)$ is the prior probability, and describes that the nature is in state w_j. The posterior probability $P(w_j|\boldsymbol{x})$ can be computed from $p(w_j|\boldsymbol{x})$ by Bayes formula:

$$P(w_j|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|w_j)P(w_j)}{p(\boldsymbol{x})} \tag{52}$$

with the evidence

$$p(\boldsymbol{x}) = \sum_{j=1}^{c} p(\boldsymbol{x}|w_j)P(w_j) \tag{53}$$

Suppose that we observe a particular $\boldsymbol{x}$ and that we contemplate taking action α_i. It the true state of nature is w_j, by definition we will incur the loss $\lambda(\alpha_i|w_j)$. Because $P(w_j|\boldsymbol{x})$ is the probability that the true state of nature is w_j, the expected loss associated with taking action α_i is merely (Duda 2001)

$$R(\alpha_i|\boldsymbol{x}) = \sum_{j=1}^{c} \lambda(\alpha_i|w_j)P(w_j|\boldsymbol{x}) \tag{54}$$

The action α_i in this approach is the decision to assign the component j to the part if $i = j, \quad j = 1, \dots, c, \quad i = 1, \dots, n$ $(n = c + 1)$ and assign the unknown component if $i = c + 1 = n$.

For recycling of precious metals the loss function can be based on the concentration of a specific precious metal in a component whereas the loss function for components with a height amount of precious metal is generally smaller than the loss functions for components with a small amount of precious metals. That would have the consequence that components…

In this approach the zero-one loss function is used:

$$\lambda(\alpha_i|w_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases} \quad i, j = 1, \dots, c \tag{55}$$

The decision process is based on discriminant functions $g_i(\boldsymbol{x})$, $i = 1, \dots c$. For the general case with risk, we can let $g_i(\boldsymbol{x}) = -R(\alpha_i|\boldsymbol{x})$ because the maximum discriminant function will correspond to the minimum conditional risk. On decision level the vector $\boldsymbol{x}$ is assigned to class w_i if

$$g_i(\boldsymbol{x}) > g_k(\boldsymbol{x}) \quad for\ all\ k \neq i \tag{56}$$

The discriminant function can be simplified by the fact that we can always multiply all the discriminant functions by the same positive constant or shift them by the same additive constant without influencing the decision. More generally, if we replace every $g_i(\boldsymbol{x})$ by the $f\big(g_i(\boldsymbol{x})\big)$, where $f(\cdot)$ is a monotonically increasing function, the resulting classification is unchanged (Duda 2001).

$$g_i(\boldsymbol{x}) = -R(\alpha_i|\boldsymbol{x}) = -\sum_{j=1}^{c} \lambda(\alpha_i|w_j) P(w_j|\boldsymbol{x}) \tag{57}$$

$$g_i(\boldsymbol{x}) = -\sum_{j \neq i} P(w_j|\boldsymbol{x}) = 1 - P(w_i|\boldsymbol{x}) \tag{58}$$

$$g_i(\boldsymbol{x}) = P(w_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|w_i)P(w_i)}{\sum_{j=1}^{c} p(\boldsymbol{x}|w_j)P(w_j)} \tag{59}$$

The discriminant function can be simplified to

$$g_i(\boldsymbol{x}) = p(\boldsymbol{x}|w_i)P(w_i) = \ln p(\boldsymbol{x}|w_i) + ln\, P(w_i) \tag{60}$$

The prior probability $P(w_j)$ describes the probability that a randomly drawn part from the entire set of components is from component j for $j < c$ and the part is an unknown component for $j = c$. To estimate these values the distribution of components over all PCBs has to be determined. In this approach the prior probabilities $P(w_j)$ were equally distributed over all c component classes with probability $P(w_j) = \frac{1}{c}$.

- Prior problem

$p(\boldsymbol{x}|w_j)$ is the state-conditional probability density function for $\boldsymbol{x}$ conditioned on w_j being the true state of nature. Under the assumption that the soft-outputs from the classifier level fusion are statistically independent the joint probability density function is equal the products of the state-conditional probability density function of one feature from the feature vector.

- Explain independence

$$\boldsymbol{x} = x_1, \dots, x_d \tag{61}$$

$$p(\boldsymbol{x}|w_j) = p(x_1, \dots, x_d|w_j) = p(x_1|w_j) * \dots * p(x_d|w_j) \tag{62}$$

This density functions are approximated by gamma distributions based on the test data outputs from the classification level.

The gamma distribution models sums of exponentially distributed random variables. The gamma distribution is defined as

$$f(x|a,b) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{\frac{-x}{b}} \tag{63}$$

where $\Gamma(\cdot)$ is the gamma function (Mathworks-Gamma 2014).

- Why gamma?

The gamma function is an extension of the factorial function with its arguments shifted down by 1, to real and complex numbers. The gamma function for all complex numbers is defined via a convergent improper integral:

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx \tag{64}$$

(Wikipedia-Gamma 2014).

The state-conditional probability density functions for the DIP14, DIP16 and Resistor network 1206 conditioned on the DIP14 component based on the test data are shown in…

$$p\left(x_i = DIP14 \middle| w_j = DIP14\right) = \tag{65}$$

$$p\left(x_i = DIP16 \middle| w_j = DIP14\right) = \tag{66}$$

$$p\left(x_i = \text{Resistor network } 1206 \middle| w_j = DIP14\right) = \tag{67}$$

Asf

3.7 Optical character recognition of electronic component marking

The optical character recognition (OCR) of printed text is widely studied and used in numerous applications like book scanning for digitalization, data entry for business documents, check and passport or license plate recognition. The automatic inspection of IC markings is a field which mainly focuses on inspection and quality control of PCB assembly processes. Inappropriate placement of chips and surface mounted devices (SMDs) can automatically be detected and corrected (B. Luo, Y.Gao, Z.Sun 2013). This approach is focusing on the inspection of IC markings whereas makings of other component like capacitors or coils are out of focus because of their complexity.

3.7.1 Optical character recognition difficulties

The difference between the inspection of IC markings of PCB assembly line lies in the quality of the ICs and there markings. Newly printed IC markings have much better quality than markings from ICs which can be found in electronic scrap. The following difficulties of the optical character recognition of IC markings are caused by the fact that the ICs are from PCB scrap but also universal for similar OCR tasks.

- Company logos or symbols in character lines
- Symbols for part orientation confuse OCR software
- Dirty disturb segmentation process
- Scratches disturb segmentation process
- Broken characters of IC markings
- Overwritten characters
- Skew IC markings
- Scraped IC markings

- Different character fonts and character size
- Uneven illumination based on shadows from height components beside the examined component

Some difficulties about IC marking recognition from electronic scrap are shown in Figure 39.

Figure 39: Difficulties of IC marking recognition

3.7.2 Optical character recognition flow chart

- Assumption made for the algorithm (character size, baselines,…)

The most important step of this OCR approach is the character recognition step where the binarized image of characters is mapped to the recognized ASCCI characters. Therefore the two OCR programs “Tesseract” and “Cognex Vision Pro” were used and compared based on the Electronic component marking recognition problem. The software Tesseract was already used in mobile IC Package Recognition (Patrick Blaes, Chris Young n.d.). For OCR engines without a-priori knowledge about the OCR task it is pretty difficult to identify electronic marking. To get a suitable recognition result the preprocessing steps in the flow chart in Figure 41 were carried out.

Each IC component has different requirements for the IC marking recognition algorithm what does it matter that the markings have different properties. Properties which have to be known for the algorithm and are stored in the component database are the region of interest (ROI) for the IC marking and the subset of characters making up the marking. For the SMD resistor 1206 component for example the character subset could be {“0”, “1”, “2”, “3”, “4”, “5”, “6”, “7”, “8”, “9”, “R”} because smaller character subsets increases the recognition rate. The marking recognition flow chart is shown in Figure 41.

The input of the process is the already recognized part image. At first the OCR-ROI is selected from the part image to reduce the character search space and cut component solder joints and component boundaries. The RGB-image is converted in grayscale image caused by the fact that the characters are white (bright) and the character background is black (dark). Median filtering is applied to reduce noise mainly salt and pepper noise.

To emphasize the characters of the markings a Laplacian of Gaussian (LoG) filter is applied. The LoG kernel is a rotationally symmetric filter which is mainly used for edge detection. The filter is composed of the second derivative (Laplace operator) of a Gaussian filter shown in equations (69) and (68). The approximated discretized kernel mask is of size h x h where h is in pixel. In this approach the kernel size is changing linear with the image scale so that the kernel mask size is constant 1mm ($h = imagescale\ [pixel]$) and in practice lies between 50 and 120 pixels. The standard deviation of the Gaussian is constant $\sigma = 0.5$.

- Explane LoG operator

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{68}$$

$$\nabla G(x,y) = \frac{\partial^2 G(x,y)}{\partial x^2} + \frac{\partial^2 G(x,y)}{\partial y^2} \tag{69}$$

The next step is the blob segmentation which is done by Otsu's segmentation method (OTSU 1978). Otsu method is a segmentation process based on a global segmentation threshold which is computed by minimizing the intra-class variance (variance within classes). After segmentation step a morphologic closing operator is applied to reduce holes in the character blobs. The size of the rectangular closing kernel (h x h) changes linear with the image scale $h = [0.05 * imagescale)[pixel]$.

After the segmentation process blobs that do not correspond to a character still exist. Therefore the areas of the blobs are estimated and blobs with an area smaller $area_{min}$ and blobs with an area greater $area_{max}$ are rejected. The next step is the rough determination of possible lower character baselines. The y coordinate of the lower right corner of the blobs bounding box is used as samples to find upper baselines. This is done by computing the probability density estimate which is done with the Matlab function ksdensity. The function returns a probability density estimate for samples based on a normal kernel function and is evaluated at equally spaced points that cover the range of the data (ksdensity, mathworks 2014). In this approach 1000 equally spaced points from zero to one are used whereas the samples are normalized by the height of the image, the smoothing parameter σ is set to 0.025. All local maximums in the probability density function are potential lower character baselines.

After demining potential character baselines the blobs are assigned to the baselines based on the distance threshold $distance_{char,\ potential\ baseline} = 0.25mm$. All characters that distance from baseline is shorter than $distance_{char,baseline}$ are assigned to the baseline as potential characters of the baseline. To remove manufacturer symbols or dirt that are segmented as potential characters, baselines with a number of assigned blobs less than or equal two are removed together with their assigned blobs. This assumption is based on the condition that part names usually consist of more than two characters.

To remove blobs that correspond to a baseline but are no characters the RANSAC outlier detection approach is used to estimate baseline models and select all characters that fit the baseline model with a distance error from the baseline smaller $distance_{char,baseline} = 0.1mm$. This is done with the lower and upper baseline of the character lines.

- RANSAC

Once again baselines with a number of assigned blobs less than or equal two are removed together with their assigned blobs.

In the next step the characters which are assigned to baselines are segmented in character lines (words). These words are transferred as an image to the character recognition software Tesseract or Cognex Vision Pro. The output of these software are the recognized characters of the word image. A comparison of the two OCR programs Tesseract and Cognec Vision Pro is done in . The settings and difficulties of the software is mentioned in **Fehler! Verweisquelle konnte nicht gefunden werden.**. The recognized words are composed to labels whereas each label is a potential part name.

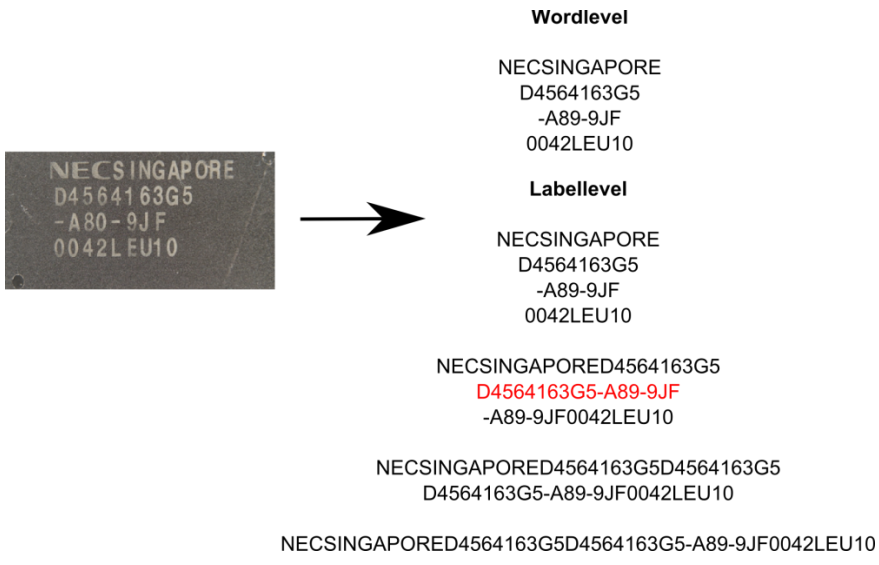


Figure 40: Label composition from words

Potential part names are requested by the Octopart API (www.Octopart.com) by sending the composed labels. After making a label request, the Octopart API sends back a list of potential part names located in their database which could correspond to the requested label. The distance between the potential part names and the requested label is determined. The distance measure is the levenshtein distance which assigns a distance to two words based on their similarity. This is done with all labels of the marking and the potential part name with the smallest distance to requested label is assigned as part name to the part.

- levenshtein distance

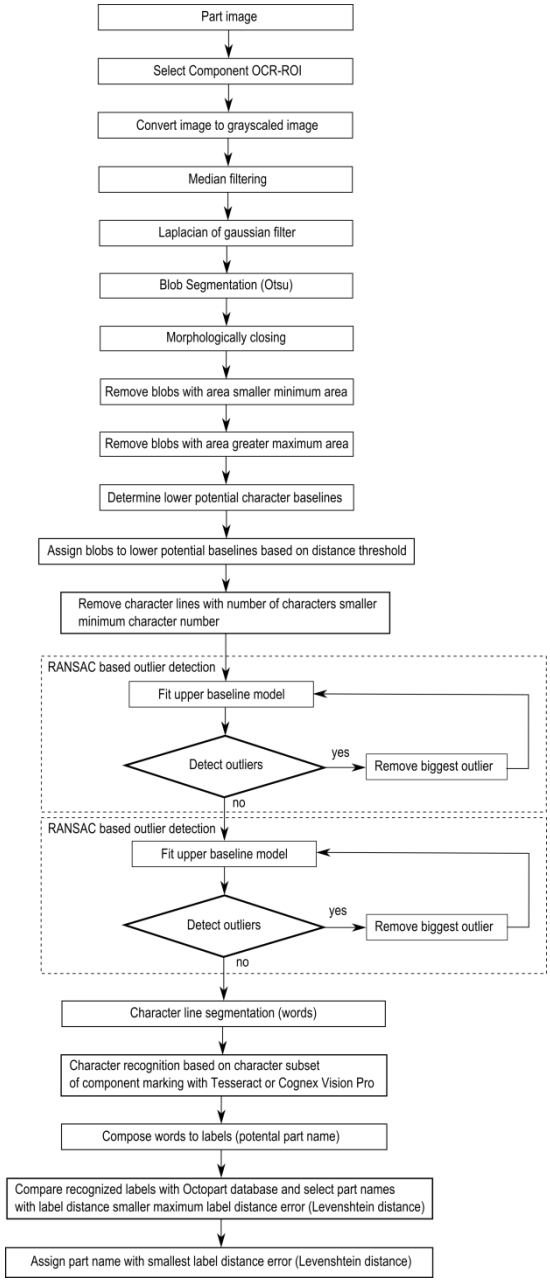


Figure 41: IC marking recognition flow chart

asd

3.7.3 Tesseract OCR engine

Tesseract is an open-source OCR engine that was developed by HP between 1984 and 1994. The program is written in C and C++ and can be used on varies platforms. Since 2006 Tesseract

development was sponsored by Google and provides support for various languages. A comparison between Tesseract 3.0.1 and FineReader10 Corporation Edition from ABBYY shows that there is no significant difference in accuracy between both software engines. The differences in accuracy depend on quality and font of the characters whereas each engine has its advantages and disadvantages (Helinski 2012).

For character recognition with Tesseract, the markings were decomposed in lines referring to the flow chart in Figure 41. The segmented binarized character line images where transferred to tesseract engine by the command-line interface in Matlab and the recognized results were stored in a text file. Tesseract is an already trained OCR engine and therefore can directly be used without training. The following settings were made to improve the accuracy rate.

- Character limitation subset was set to "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ/"
- Tesseract pagesegmode: 7 = Treat the image as single text line

3.7.4 Cognex VisionPro® OCRmax engine

OCRMax™ is a font-trainable OCR and OCV (Optical character recognition and Optical character verification) tool from the Cognex VisionPro® software suite for image processing (VisionPro 2014). In this approach the OCR engine OCRMax™ was just like Tesseract used to recognize characters from segmented binarized character line images. The main difference between the OCRMax™ engine and Tesseract is the fact that OCRMax has to be trained before it can be used. Therefore a training data set was composed consisting of electronic part markings. The Software was trained with 1500 characters from 84 IC markings. The following settings were made to improve the accuracy rate.

- Character limitation subset was set to "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ/"
- Einstellungen beschreiben

3.7.5 Electronic part label verification based on Octopart database

Octopart is a company that offers an electronic part database with structured data for more than 30 million parts. The Octopart tools facilitate to search parts across thousands of suppliers. An easy way to access the database is the Octopart API which provides information about up-to-date pricing and availability information, datasheets, compliance documents and technical specs for electronic components from distributors and manufacturers. Octopart allows access to information from more than 100 distributors including Digi-Key, Mouser, Newark, Premier farnell, Arrow, RS Component, Future electronics, Grainger and many others (Octopart 2014).

This tool was used for part name verification in which the recognized labels from OCR engines (Tesseract, OCRMax) were requested to the Octopart API. The response of the API is a list of equal or similar written part names provided from different suppliers. To assign a part name from the obtained list to the recognized label, the Levenshtein distance between the part names and the requested label is computed. The part name with the smallest distance less than or equal the distance threshold $distance_{label,\ oct,\ tresh} = 2$ is assigned to the part. The requests were made with the data transfer tool curl in Matlab.

- Curl, words, labels
- Analyse requested results
- Explaine TP,FP,…
- Error by manual labeling
- part level

4. Life-cycle inventory analyses of printed circuit boards

4.1 Printed circuit board region classification based on electronic part recognition results

- PCB support material (epoxy)
- Detected and not correctly classified electronic parts
- Detected and correctly classified electronic parts
- Detected,correctly classified and label recognized electronic parts

4.2 Printed circuit board LCI model

Life cycle inventory (LCI) is a process of quantifying energy and raw material requirements, atmospheric emissions, waterborne emissions, solid wastes, and other releases for the entire life cycle of a product, process, or activity (Curran 2006). An LCI is the basis of an Life cycle impact assessment (LCA) to evaluate comparative environmental impacts or potential improvements. With respect of reuse and recycling an LCI can assist organizations in comparing products or processes and considering environmental factors in material recycling. The "Guidelines for Assessing the Quality of Life Cycle Inventory Analysis" (Lynda Wynn, Eugene Lee 1995) provides a framework for performing an inventory analysis. Four steps are defined for making a life cycle inventory:

1. Develop a flow diagram of the process being evaluated
2. Develop a data collection plan
3. Collect data
4. Evaluate and report results

4.2.1 ILCD interface for automatic generation of LCI-models from PCBs

The International Life Cycle Data System (ILCD) has been developed by the Joint Research Centre - Institute for Environment and Sustainability (JRC-IES) of the European Commission to provide guidance for consistent and quality assured Life Cycle Assessment data and studies (Europen Commission 2011). The ILCD Data Format was developed for storing and structuring data set information within a data stream or file to enhance the availability of consistent and quality assured Life Cycle Inventory (LCI) data sets. It was designed to serve as reference format and for data exchange between varieties of Life Cycle impact assessment (LCA) software. The ILCD data format has been released in 2009 and has already seen some adoption among tools like GaBi or OpenLCA and databases in the meantime. The ILCD format is based on an Internet-aware, linked data approach. The ILCD format provides currently seven data set types which identify different semantic concepts in LCA modelling that are linked together via typed links called global references (Marc-Andree Wolf 2011). These types of data set concepts are:

- Process: Modelling unit and aggregated processes and result sets. Input and Output flows are modeled by global references to other datasets of type flow.
- Flow: Describes an elementary, product or waste flow. It reference one or more Flow properties.
- Flow Property: Describes physical or other properties of a flow that can be used to quantify it, for example mass. Each instance references one Unit Group data set.
- Unit Group: Describes a group of convertible units and the conversion factors to its reference unit
- LCIA Method: Describes an LCIA method and its characteristic factors
- Source: Represents an external source of information, such as literature or a database or data format. It can reference a contact it is related to.
- Contact: describes a person or organization.

- Flowpropertie – costs
- Flow to process -> flowproperties to flows
- Two different models

4.2.2 PCB flow diagram

The LCI-model in this approach is a generalize model for Printed circuit boards. It is developed to handle PCBs from scrap automatically. The quality of the LCI results depends on the composition of the specific PCB and is described in more detail in . In this work two different models are created. The first model represents the LCI model of the PCB and uses full aggregated data to quantify energy and raw material requirements, emissions, solid waste and other releases.

- Erklärung keine richtiges LCI-model
- Assembly line modellierem

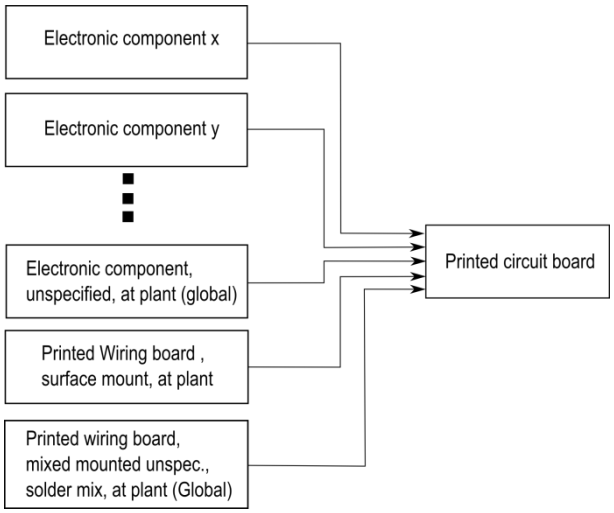


Figure 42: PCB flow diagram for LCI-model

The flow diagram for a PCB is shown in Figure 42. The PCB is consists of four different PCB regions:

- Gabi check Bestückung modellieren (system boundyries)
-

1. Detected part, recognized packages: Modeled by the Electronic component x, Electronic component y and so on.
2. Detected part, unrecognized package: Modeled by the Electronic component unspecified, at plant (global)
3. Recognized printed wiring board surface: Modeled by Printed wiring board surface mount, at plant
4. Unknown PCB regions: Modeled by Printed wiring board mixed mounted unspec., solder mix, at plant (global)

The second model represents the material composition of the PCB. This model is of interest for recycling organizations to analyze the content of precious metals or other valuable resources. Moreover the amount of hazard materials in the specific PCB can be analyzed and specially treated. The flow diagram of the PCB composition model is shown in Figure 43. The flows in the figure between the PCB components and the Materials are symbolic and depend on the content of the composition of the components.

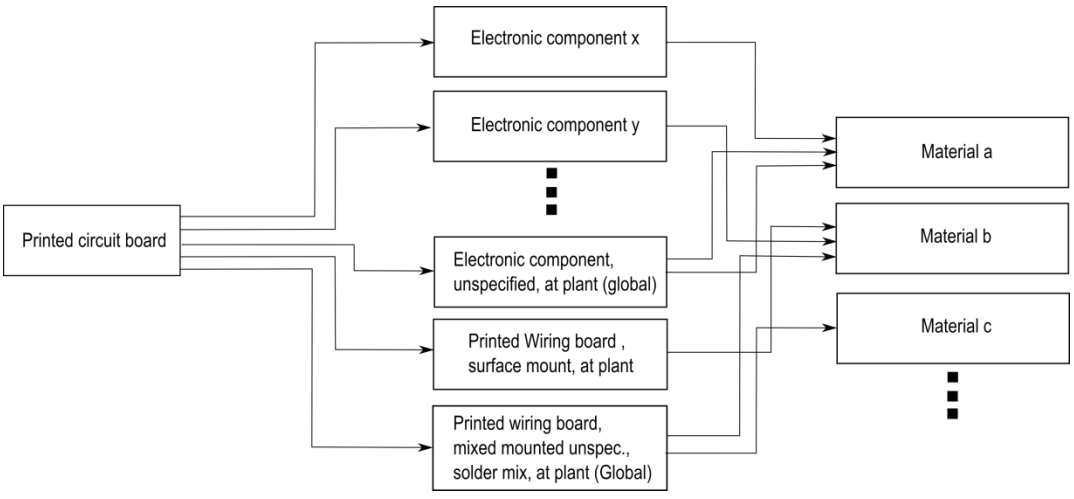


Figure 43: PCB flow diagram for composition model

The four PCB regions for a sub image of the Arduino Due board are shown in Abbildung 1. The red colored regions are parts which are detected with recognized packages (SOT223, Resistor

network, and Button). The green colored parts that were detected but the package could not be recognized. The yellow colored regions are unknown PCB regions ($A_{PWB,unknown}$) and the blue colored region is the printed wiring board surface region (A_{PWB}).

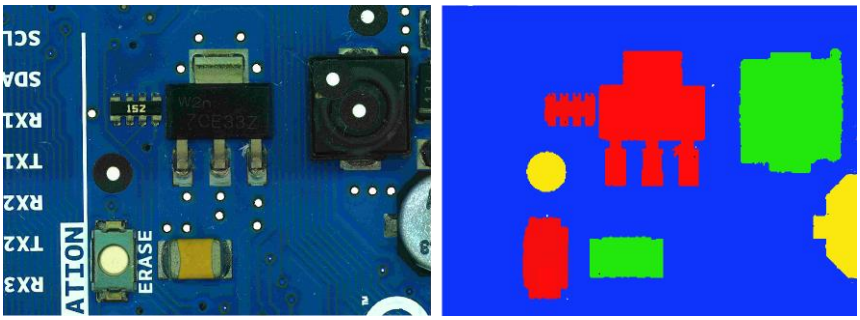


Abbildung 1: PCB regions

4.2.3 Data collection plan and data collection

The data collection is mainly based on the GaBi Extension database XI: Electronics from PE INTERNATIONAL. For each electronic component in the recognition database an ILCD package model is assigned.

- flowproperties

Detected and correctly classified electronic parts are modeled by ILCD component models. If the ILCD component model exists in the GaBi database it is used in the recognition database. If a component is not modeled in the GaBi database but a similar model which differs mainly in size, the amount of the component is scaled by mass and assigned to the component in the recognition database see formula (70).

$$N_{component,model} = N_{component,PCB} * \frac{m_{component,PCB}}{m_{component,GaBi}} \quad (70)$$

Electronic parts which are detected but the package could not be classified correctly are modeled by the “Electronic component, unspecified, at plant (global)” woher? and assigned to the component in the recognition database.

PCB regions where PCB support material (epoxy) was detected are modeled by “Printed wiring board surface mount, at plant”. The amount unit is mass and is calculated by the region area recognized in the image and the basis weight. The basis weight $w_{PWB} = 3{,}92\frac{kg}{m^2}$ is based on the information on http://www.leiton.de (Leiton: leiton-tools-gewichtsberechnung 2014).

$$N_{PWB,model} = w_{PWB} * A_{PWB} \tag{71}$$

$A_{PWB} - Printed\ wiring\ area,$

$N_{PWB,model} - Printed\ wiring\ amount$

PCB regions where an assignment of a component or PCB support material could not be made are modeled by “Printed wiring board mixed mounted unspecific, solder mix, at plant (global)”. The amount is calculated based on the basis weight $w_{PWB,mounted} = 0{,}75\frac{g}{cm^2} = 7{,}5\ \frac{kg}{mm^2}$ which was determined by the average value of 25 PCBs which is listen in detail in **Fehler! Verweisquelle konnte nicht gefunden werden.**.

$$N_{PWB,mounted,model} = w_{PWB.mounted} * A_{PWB,mounted} \tag{72}$$

$A_{PWB,mounted} - Printed\ wiring\ area\ of\ mounted\ PCB\ region$

$N_{PWB,mounted,model} - Printed\ wiring\ amount\ of\ mounted\ PCB\ region$

4.2.4 Evaluation and results

The results of the two models are different in a way that the composition model quantifies the materials which make up the PCB. Parts with a high amount of precious metals or other valuable materials for recycling can be determined and detached. The separate treatment can increase the concentration of valuable materials in the separated electronic scrap and therefor is an important factor for an economic recycling process. The increase of concentration for some valuable materials is discussed in detail in chapter 0.

The LCI-model quantifies energy and raw material requirements, atmospheric emissions, waterborne emissions, solid wastes, and other releases. It can be used to discover PCB boards or electronic parts containing hazard materials that can be specially treaded.

The material composition model and the LCI-model for the Arduino Due board are drawn up in chapter 5.7.3.

5. Experimental results

5.1 Implementation

5.2 Dataset creation

The dataset consist of 12 electronic components which were analyzed. The components are listed in **Fehler! Verweisquelle konnte nicht gefunden werden.**. The component selection depends on the occurring frequency on the available printed circuit boards. It was taken care that also similar looking components were selected. Therefor the DIP14 component and DIP16 component which differ almost only by number and position of solder joints were selected. In addition the tantalum capacitors of different size but similar appearance were selected. For electronic component recognition, a machine learning application was used whereas multiple representation of the component must be created to analyze representative features. The component representations are taken from different parts of a component and different printed circuit boards to create a representative dataset. The available printed circuit boards are seen in .

To detect the edges of the part border, border pixels are also selected from the printed circuit board images as can be seen in Figure 44. Additional important information and properties of the component are listed in Table 2.

Table 2: Component properties

Component properties	Description
Package properties	
Component length	
Component width	
Component border size	
Package DOF	
OCR properties	
ROI for optical character recognition	
Subset of characters for optical character recognition	
Maximum and minimum number of OCR lines	
Frequency features generation properties	
Image scale for frequency feature generation	
Number of Fourier coefficient features	
Border cut information	
Color histogram features	
Image scale for histogram feature generation	
Segment features	
Image scale for histogram feature generation	
Number of initial seed points for region growing approach	
PCA reconstruction features	
Image scale for histogram feature	

generation	
Kernel size for LoG (Laplacian of Gaussian) edge detection	
Number of PCs	
LCI properties	
ILCD-model full aggregated model	
ILCD-model composition model	

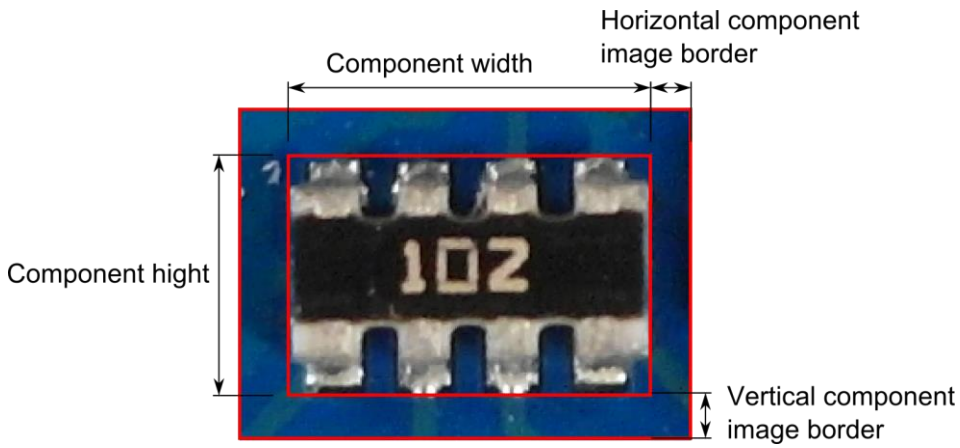


Figure 44: Component border definition

A section of the component database is shown in Figure 45.

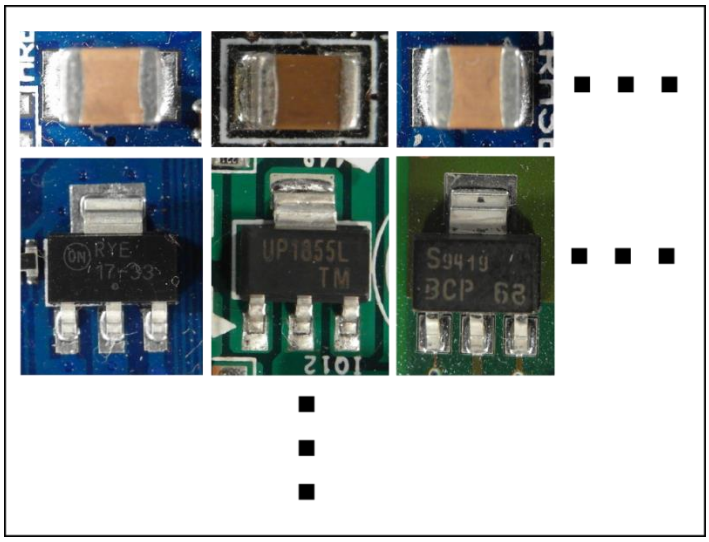


Figure 45: Database section

5.2.1 Image acquisition

The image acquisition was done with a Samsung EX2F camera and a working distance in a range from 20 mm to 120 mm through the Object. Autofocusing was used to get sharp images. The working distance was adapted to the size of the component in which the distance was decreased for smaller components and increased for bigger components. For illumination a bright-field incident illumination was selected because it generates a uniformly bright, well-contrasted image (Imaging 2012). The lighting sources consist of four DSL-1110 table lamps with diffusion film to generate a uniformly bright and diffuse illumination. The image acquisition system is seen in Figure 46.

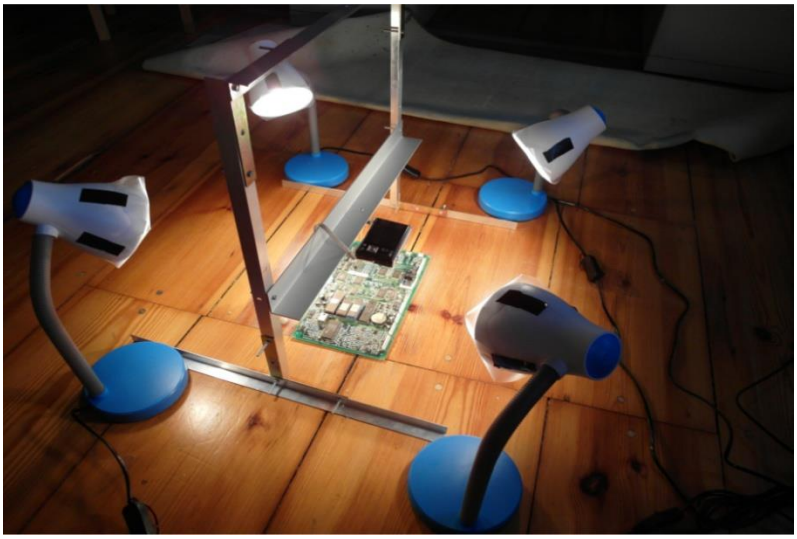


Figure 46: Image acquisition system

- Verzerrungen in bildern -> ausschnitt
- Schatten
- Winkel
- Kamera bewerten -> ausblick besseres kamerasystem
- Tiefenschärfe
- Telezentrisches Objektiv

5.2.2 Dataset composition

The dataset used in the experiments consist of 2000 parts from 12 component classes. The dataset composition is shown in Table 3.

Table 3: Dataset composition

	Number of component images	Number of training data	Number of test data
Tantalum capacitor			
SMD Aluminum electrolytic capacitor			
QFP100			
SMD Resistor Network array 1206, 4 Resistors			
SMD Transistor SOT23-3			
DIP14			
DIP16			
SMD Resistor 1206			
SOIC-8			
Ceramic capacitor 1210			
SOT223-3			
SMD Resistor 0806			
TO263			
Quartz HC-49/S			

5.3 PCB surface detection results

The process of PCB surface detection based on the PCB surface color is explained in chapter 3.2.2.

5.4 Feature selection results

The out-of-bag error depends on the number of random forest trees. The oob-error depending on the number of trees for 3136 FFT features extracted from the Resistor network 1206 component was computed. The red graph shows the out-of-bag error from the two step feature selection (FS+FR), the blue one the out-of-bag error from the random forest feature selection (RF) and the green one the out-of-bag error from fisher score (FS) feature selection with 235 selected features. The graphs show that the error rate of the FS+RF feature selection approach decreases faster and becomes smaller compared to the others whereas the oob-error does not show a big difference between the algorithm what indicates that the samples tend to be well linearly separable.

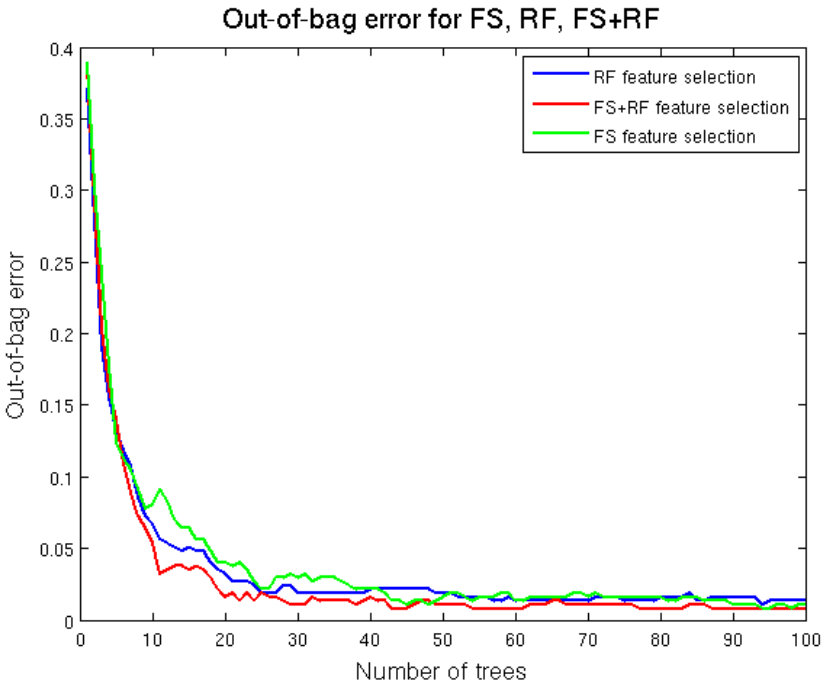


Figure 47: A comparison of different feature selection approaches

In this approach the feature selection algorithm based on Fisher score and Random forest described in **Fehler! Verweisquelle konnte nicht gefunden werden.** was used to select a subset of important features for classification. The most important features depend on the component therefore feature selection was applied to each component dataset. A list of the most important features from all examined components is shown in Appendix A. The numbers in the

table correspond to the feature numbers of the associated feature set. The assignment of the features numbers to the features is explained in .

Several selected important features are examined in detail to understand and confirm their importance for specific components.

5.4.1 Fourier features

The second most important feature of the SMD Resistor Network array 1206 is the second Frequency feature. The feature is the real part of the frequency coefficient with period of image high. It is the amplitude of the cosine transform in vertical direction. The mainly black region in the middle of the resistor is clearly visible. Toward the vertical image border the intensity becomes brighter caused by the reflective solder joints. This intensity gradient is typical for the resistor network and the curve correspond to the cosine curve of the second frequency feature. The elementary image of the frequency is shown in Figure 48.

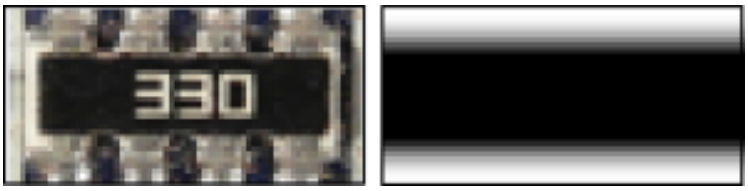


Figure 48: Resistor network 1206 and the most significant real part elementary image

The values have been linearly scaled to vary between 0 (black and 255 (white).

5.4.2 Color features

The most important feature of the tantalum capacitor is a color feature which makes sense under the knowledge that the tantalum capacitor is a yellow-orange colored component and very different from the colors of other components or image regions in the PCB image.

- Bild + erläuterung

5.4.3 Segment features

The second most important feature of the Ceramic capacitor 1206 is the seventh segment feature. The seventh segment feature is the vertical component of the center of gravity from the segment which was produced by the region growing approach with the seed point at the seed position y=1.7mm, x=0.26mm. The brown/orange segment in the middle of the capacitor is significant for the component. Compared to other components the probability that a seed point located near the image border produces a segment with the center of gravity in the middle of the image is much smaller. The red marker shows the seed point of the segment which was produced by the region growing approach. The blue marker is the center of gravity from the segment. The vertical component of the center of gravity is the second most important feature for the ceramic capacitor.

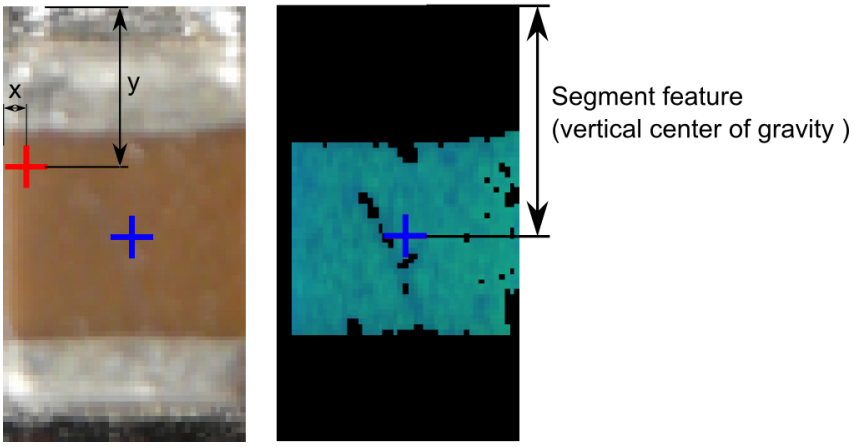


Figure 49: Most important segment and seed point from ceramic capacitor

5.4.4 PCA reconstruction feature

The most important feature of the SMD Aluminum electrolytic capacitor is the PCA-reconstruction feature. That can be explained by looking at the circular border of the cylindrical part. The rounded border reflects the light almost independent from the beam angle of the illumination. That forms a bright shiny circle that is striking in the Laplacian of Gaussian (LoG) filtered images and can be efficiently be compressed into the part image PCs. A LoG filtered

edge image of the SMD Aluminum electrolytic capacitor and the unit matrix projection into the PCs is shown in Figure 50.

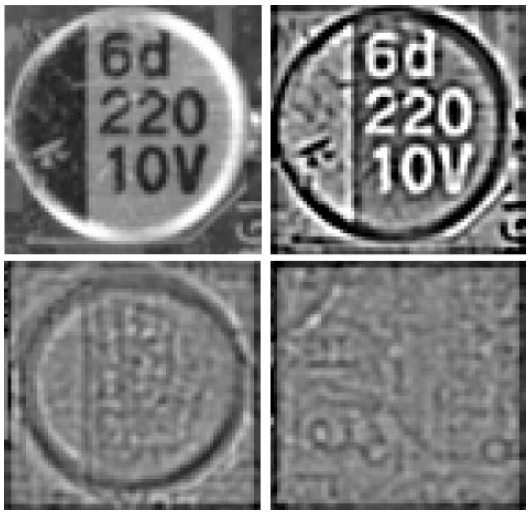


Figure 50: SMD Electrolyte capacitor (top, left), SMD Electrolyte capacitor edge image (top, right), unit matrix projection into component PCs (bottom, left), unit matrix projection into non-component PCs (bottom, right)

5.4.5 Dependence of classification accuracy from number of selected features

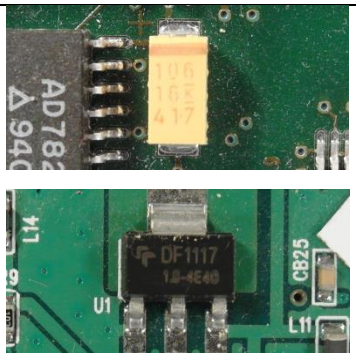
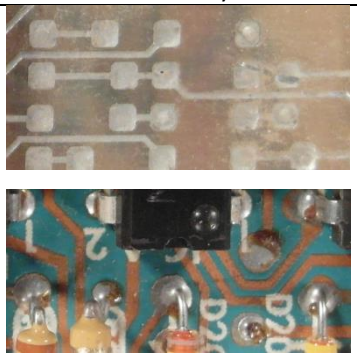
5.5 Classification results

The One-vs.-rest classification strategy is based on the approach that for each component a classifier is trained and tested. The training set and test set consist of part images and non-part images.

There are two approaches to select the non-part images whereas the first approach is based on the idea that the part detection algorithm works in a way that the algorithm detects almost all parts in the PCB image and that most of the parts are in the database. Under these requirements the non-part images consist of images from parts of different components. The second approach is based on the idea that the non-part images should represent a plurality of

possible images and therefore the non-part images are arbitrary selected image sections from the PCB images. An example of both approaches for the DIP14 component is shown in Table 4.

Table 4: Dataset approaches for non-part images

Part images for DIP14	Non-part images for DIP14 (images from different parts)	Non-part images for DIP14 (images from arbitrary image section)
		

Both approaches use the same number of part images and non-part images and have advantages and disadvantages with respect to the representativeness of the data. If the non-part images consist of only images from different components than the variance of the non-part image set is smaller and the accuracy should be better greater. On the other hand for the classifier it is more difficult to handle non-part images from components that were not in the training set or images on which no part is seen.

- bbox
- continue

5.5.1 Random forest classifier results

Five random forest classifiers were trained whereas the first to fourth are based on the four selected feature sets from Frequency features, Color features, Segment features and PCA-reconstruction features which were extracted from the four feature extraction algorithms

described to chapter 2.2. The accuracy for the random forest classifier from the mean accuracies of all twelve components is shown in Table 5. A detailed breakdown can be found in **Fehler! Verweisquelle konnte nicht gefunden werden.**.

The used random forest parameters are as followed:

- Number of random forest trees: 100

Table 5: Random forest classification results

		Frequency features	Color features	Segment features	PCA reconstruction features	Features selection from all feature sets
Average recognition accuracy of all Components	True positive					
	False positive					

The results for the One-vs.-rest classification strategy is shown in .

5.5.2 Support vector machine classifier results

Sdf

5.5.3 Multiclass classification result

dfg

5.6 Optical character recognition results

To evaluate the optical character recognition results different recognition levels are defined. The lowest one is the character level where

- continue

5.6.1 Optical character recognition dataset and limits

The optical character recognition dataset consists of 85 ICs which were acquired with an image resolution of $100\ pixel/mm$. All components were manually labeled according to the accuracy level scheme in explained in 3.7.2.

To refine the investigation of Optical character recognition of IC markings the following restriction limits were taken.

- The components which are used to investigate the optical character recognition of IC markings have a black (dark) surface and the markings are white (bright).
- Marking characters have a minimum height of 1.0 mm
- Makings made by laser engraving are out of focus
- The IC markings have to be readable by humans

Parts that are out of that restriction are not used in the OCR dataset for IC marking inspection.

5.6.2 Optical character recognition accuracy result on character level, word level and label level

To evaluate the optical character recognition results the labeled component markings and the recognized marking with OCR software are compared on different accuracy levels (Helinski 2012).

The character level accuracy of the OCR engine recognition is calculated as follows:

$$A_{c,i} = 1 - \frac{e_{c,i}}{c_i} \tag{73}$$

were $e_{c.i}$ is the number of character errors (insertions, substitutions and deletions) of the component marking i and c_i is the number of all characters of the marking i. The average character level accuracy over all 85 component markings is calculated as follows:

$$A_c = 1 - \frac{\sum_{i=1}^{85} e_{c,i}}{\sum_{i=1}^{85} c_i} \tag{74}$$

The word level accuracy of the OCR engine recognition is calculated as follows:

$$A_{w,i} = 1 - \frac{e_{w,i}}{w_i} \tag{75}$$

were $e_{w.i}$ is the number of word errors of component marking i and w_i is the number of words of component marking i. The average word level accuracy over all 85 component markings is calculated as follows:

$$A_w = 1 - \frac{\sum_{i=1}^{85} e_{w,i}}{\sum_{i=1}^{85} w_i} \tag{76}$$

The accuracy results of the OCR engines Tesseract and OCRMax on all accuracy levels is shown in Table 6.

Table 6: OCR accuracy results

	Tesseract	OCRMax
Character level accuracy A_c	1352/1704 (79.3%)	1342/1704 (78.8%)
Word level accuracy A_w	123/234 (52.6%)	126/234 (53.9%)

The label level accuracy was not studied because of the height number of non-part labels with many characters which would result in a height error rate and is not representative caused by the fact that labels have to be filtered based on a part-name database. An investigation of the accuracy on label level with the Octopart database is done in chapter 5.6.3.

5.6.3 Octopart based part name assignment

The online electronic component database Octopart gives the opportunity to verify recognized part markings. The OCR dataset was used to test the assignment of recognized markings to components in the Octopart database. Therefore the labeled markings where decomposed in words (word-level) and the words where composed to labels (label-level). The words and labels where requested with the Octopart-API and the results were analyzed according to the method in chapter 3.7.5. One of the classes “part-name” and “non-part-name” is assigned to each of the words and labels. The analyzed results were evaluated according to the assignment. The words/labels that are part-names and are assigned to the right part in the Octopart database are true positive labled results. Words/labels that are non-part-names like manufacturer

names, country of manufacture, production numbers etc. and which are not assigned to parts in the Octopart database are labeled as true negative. Words/labels that are not part names but assigned to parts in the Octopart database are labeled as false positive. Words/labels that are parts but are not assigned to parts in the Octopart database or assigned to wrong parts in the database are labeled as false negative. The confusion matrix for the manual labeled words and labels are shown in Table 7.

Table 7: Confusion matrix of the manual labeled words (word-level) verified with Octopart database

	Condition: part name	Condition: non-part name
Test outcome: part name	60/73 (82.2%)	6/161 (3.7%)
Test outcome: non-part name	13/73 (17.8%)	155/161 (96.3%)

The confusion matrix for the manual labeled words and labels is shown in Table 8.

Table 8: Confusion matrix of the manual labeled labels (label-level) verified with Octopart database

	Condition: part name	Condition: non-part name
Test outcome: part name	61/75 (82.2%)	6/395 (1.5%)
Test outcome: non-part name	14/75 (17.8%)	389/395 (98.5%)

The accuracy rate on part-level in Table 9 shows how many parts were assigned to a part in the Octopart database whereas the part names were manual labeled and verified with Octopart database on word level.

Table 9: Accuracy rate of part assignment with manual labeled parts on word level verified with Octopart database (part-level)

Part assignment true	59/85 (69.4%)
Part assignment false	26/85 (30.6%)

The accuracy rate on part-level in Table 10 shows how many parts were assigned to a part in the Octopart database whereas the part names were manual labeled and verified with Octopart database on label level.

Table 10: Accuracy rate of part assignment with manual labeled parts on label level verified with Octopart database (part-level)

Part assignment true	60/85 (70.6%)
Part assignment false	25/85 (29.4%)

Th results in Table 9 and Table 10 show that the analyses of the part names on label level increases a little the accuracy rate on part level compared to the analyses on word level.

The confusion matrices for the recognized part markings with the OCR engine Tesseract on word-level is shown in Table 10.

Table 11: Confusion matrix of the Tesseract recognized words (word-level) verified with Octopart database

	Condition: part name	Condition: non-part name
Test outcome: part name	31/73 (42.2%)	9/161 (5.6%)
Test outcome: non-part name	42/73 (57.8%)	152/161 (94.4%)

The confusion matrices for the recognized part markings with the OCR engine Tesseract on label-level is shown in Table 8.

Table 12: Confusion matrix of the Tesseract recognized labels (label-level) verified with Octopart database

	Condition: part name	Condition: non-part name
Test outcome: part name	33/75 (44.0%)	8/473 (1.7%)
Test outcome: non-part name	42/75 (56.0%)	465/473 (98.3%)

The accuracy rate on part level is shown in Table 9.

Table 13: Accuracy rate of part assignment with Tesseract OCR engine on word level verified with Octopart database (part-level)

Part assignment true	30/85 (35.3%)
Part assignment false	55/85 (64.7%)

Table 14: Accuracy rate of part assignment with Tesseract OCR engine on label level verified with Octopart database (part-level)

Part assignment true	31/85 (36.4%)
Part assignment false	55/85 (63.6%)

The confusion matrices for the recognized part markings with the OCR engine OCRMax on word-level is shown in Table 15.

Table 15: Confusion matrix of the OCRMax recognized words (word-level) verified with Octopart database

	Condition: part name	Condition: non-part name
Test outcome: part name	44/73 (60.3%)	13/161 (8.1%)
Test outcome: non-part name	29/73 (39.7%)	148/161 (91.9%)

The confusion matrices for the recognized part markings with the OCR engine OCRMax on label-level is shown in Table 15.

Table 16: Confusion matrix of the OCRMax recognized labels (label-level) verified with Octopart database

	Condition: part name	Condition: non-part name
Test outcome: part name	44/75 (58.7%)	9/473 (1.9%)
Test outcome: non-part name	29/75 (41.3%)	464/473 (98.1%)

The accuracy rate on part level with OCR engine OCRMax is shown in Table 17.

Table 17: Accuracy rate of part assignment with OCRMax OCR engine on word level verified with Octopart database (part-level)

Part assignment true	44/85 (52.0%)
Part assignment false	41/85 (48.0%)

Table 18: Accuracy rate of part assignment with OCRMax OCR engine on label level verified with Octopart database (part-level)

Part assignment true	44/85 (52.0%)
Part assignment false	41/85 (48.0%)

5.6.4 Octopart based part price assignment

To evaluate the economic sustainability of reuse of electronic part it is necessary to estimate the economic value of recognized parts. One indicator of valuable parts is the original price of the part. The Octopart database gives the possibility to request the price for a part if the part could be assigned to a part in the database. Unfortunately not all suppliers publish their prices and the therefore a price can just be assigned for a subset of the parts. The prices of all manual labeled parts were requested, and the price rate was calculated as follows:

$$A_{price} = \frac{\#parts_{price}}{\#parts_{assigned}} = \frac{30}{59} = 0.509\ (50.9\%) \tag{77}$$

where $\#parts_{price}$ is the number of parts where a price could be estimated and $\#parts_{assigned}$ is the number of parts that could be assigned to a part in the Octopart database. The price rate shows that for around 51% a part price could be estimated with the Octopart database.

To estimate the reuse potential of electronic components a critical economic value for the parts can be estimated which is a balance between the costs of desoldering the part and the costs of quality check one hand and economic value of a part on the other hand. The AutDem project (Automated disassembly of PWBs) estimates the cost for automated desoldering between 1.2 and 2.5 Euro depending on desoldering time, line configuration and utilization (Irina Stobbe, Hansjörg Griese 2002).

The maximum value of 2.3 Euro was used to estimate the critical price rate which was calculated as follows:

$$A_{price,critical} = \frac{\#parts_{price,critical}}{\#parts_{assigned}} = \frac{10}{59} = 0.17\ (17.0\%) \quad (78)$$

where $\#parts_{price,critical}$ is the number of parts where a price could be estimated and the price was greater than 2.5 Euro and $\#parts_{assigned}$ is the number of parts that could be assigned to a part in the Octopart database. The critical price rate shows that for around 17% of the assigned parts a price could be estimated which is greater than the critical price of 2.5 Euro based on the Octopart database. A detailed discussion about the reuse potential is made in chapter 0.

5.7 Life-cycle inventory analyses results

sdf

5.7.1 GaBi-Software and LCI data availability of electronic components

Saf

5.7.2 Recycling and reuse potential of electronic components

Increase of tantalum concentration in scrap by selective dismantling

Tantalum is one of the Rare earth materials (REE) which production increases every year. Around 1400 tons of tantalum is produced worldwide per year. Around 60% of the tantalum is used in capacitors for electronic equipment like Desktop PCs, Mobile phones or others. (Perine Chancerel, vera Susanne Rotter 2013).

The concentration of tantalum in electronic scrap is low and the present economic value is not very high compared to other metals like gold or palladium which it makes challenging to recycle tantalum from electron scrap. In the present recycling process is focused on the recycling of precious metals caused by the fact that the economic value is much higher compared to other materials.

The concentration of tantalum in tantalum capacitor scrap is between 35% and 50% which makes it economically attractive to recycle tantalum capacitors (Perine Chancerel, vera Susanne Rotter 2013). The approach of automatic optical inspection (AOI) for tantalum capacitor localization on PCBs and the automatically selective disassembly of the tantalum capacitors can increase the recycling rate and prevent from a worldwide lack of tantalum caused by higher production rates. A market for tantalum capacitor scrap already exists (http://tantalumrecycling.com/ 2015).

5.7.3 Arduino Due board LCI-model

The Arduino Due is a microcontroller board based on the Atmel SAM3X8E ARM Cortex-M3 CPU (Arduino 2014). The Arduino board consists of an open-source hardware design and was used as a reference board in the INPIKO-Project. The Arduino Due board was used as LCI example reference due to the fact that an open-source eagle layout already exist and a part list can be easily exported in the eagle software.

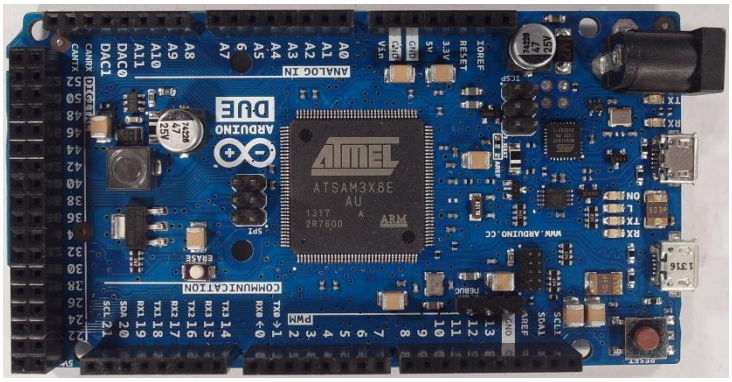


Figure 51: Arduino Due board

The Arduino Due board consists of 125 parts from 32 different components. The LCI-model (Life-cycle-inventory-model) was created based on the assumption that all parts are correctly detected, classified and all IC marking were correctly recognized. All parts are correctly assigned to the part in the Octopart database.

Each part of the Arduino Due board was modeled by an ILCD-model whereas ILCD models were exported from the GaBi Extension database XI: Electronics from PE INTERNATIONAL and scaled according to the component size. Electronic parts that could not be assigned with an associated part from the GaBi database were replaced with a replacement model. 16 of the 32 components of the Arduino Due board could be assigned to a model in the Gabi database and 17 of the 32 components had to be replaced by replacement models. The replacement models are also electronic components from the GaBi Extension database XI which were selected according to a similar structure and similar characteristics as the component. The assignments and the replacement models are listed in **Fehler! Verweisquelle konnte nicht gefunden werden.**.

The resulting process model input parts for the Arduino Due model are shown in Table 19.

Table 19: Arduino Due parts of the LCI model

Input	Amount
Diode MELF (130mg) D2.6x5.2 PE	2
Diode power DO214/219 (93mg) 4.3x3.6x2.3 PE	1
IC TQFP 100 (520mg) 14x14x1.0 PE	1
IC TQFP 32 (70mg) 5x5x1.0 PE	1

IC TSSOP 8 (28mg) 3.0x2.9x1.2 flash PE	1
Kondensator Al-Elko SMD (300mg) D6.3x5.4 PE	2
Kondensator Keramik MLCC 01005 (0,054 mg) 0,4x0,2x0,22 (Nichtedelmetalle) PE	31
Kondensator Keramik MLCC 0603 (6mg) 1.6x0.8x0.8 PE	1
Kondensator Keramik MLCC 2220 (450mg) 5.7x5x2.5 PE	7
LED SMD low-efficiency max 50mA (35mg) without Au 3.2x2.8x1.9 PE	6
Quartz Crystal (500mg) 11.05x4.65x2.5 PE	2
Schalter Tact (242mg) 6.2x6.3x1.8 PE	1
Spule Miniatur gewickelt SDR1006 (1.16g) D9.8x5.8 PE	1
Spule Multilayer Chip 1812 (108mg) 4.5x3.2x1.5 PE	7
Transistor signal SOT223 3 leads (110mg) 3.8x7.65x2.3 PE	1
Transistor signal SOT23 8 leads (18mg) 1.4x3x2 PE	6
Widerstand Dickfilm Flat Chip 0402 (0.75mg) PE	47
Widerstand Dickfilm Flat Chip 0603 (2.1mg) PE	5
Widerstand Dickfilm Flat Chip 1206 (8.9mg) PE	20

The electronic components consist of materials which can be recycled under certain circumstances. The material composition of the Arduino Due parts is shown in Figure 52.

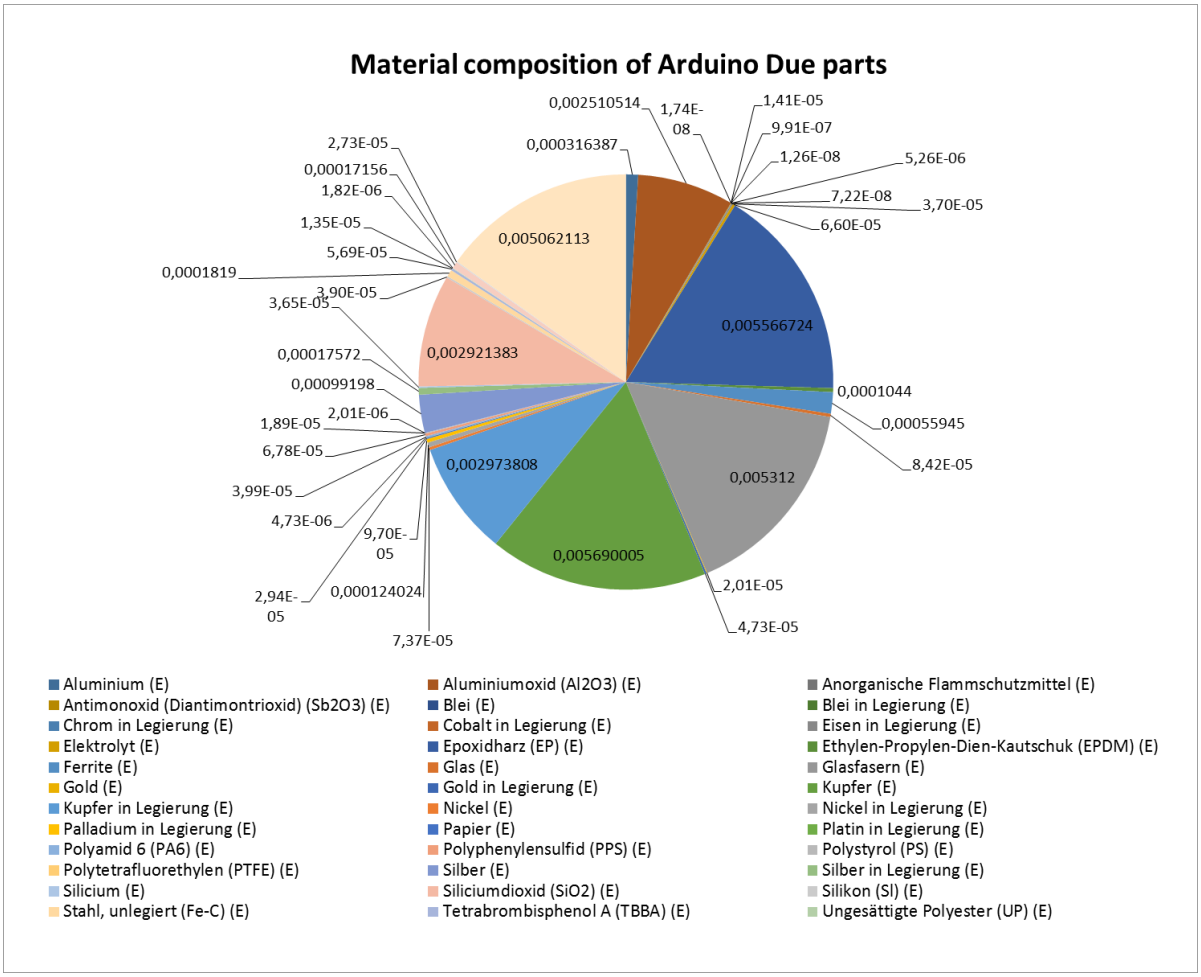


Figure 52: Material composition of Arduino Due parts [kg]

The material prices of the materials contained in the Arduino Due parts are shown in Figure 53.

Not all material prices

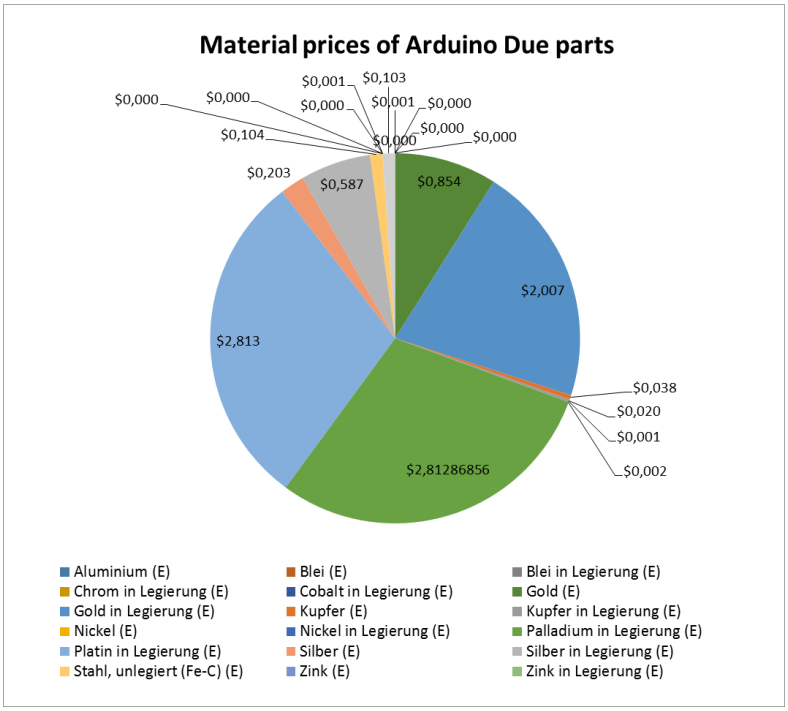


Figure 53: Material prices of Arduino Due parts

Most of the precious metals are valuable materials in PCBs which can be economically recycled. One of the most valuable materials is gold which is included in components or used as protective coating on electric connectors. The amount of gold distributed over the parts of the Arduino Due board is shown Figure 54.

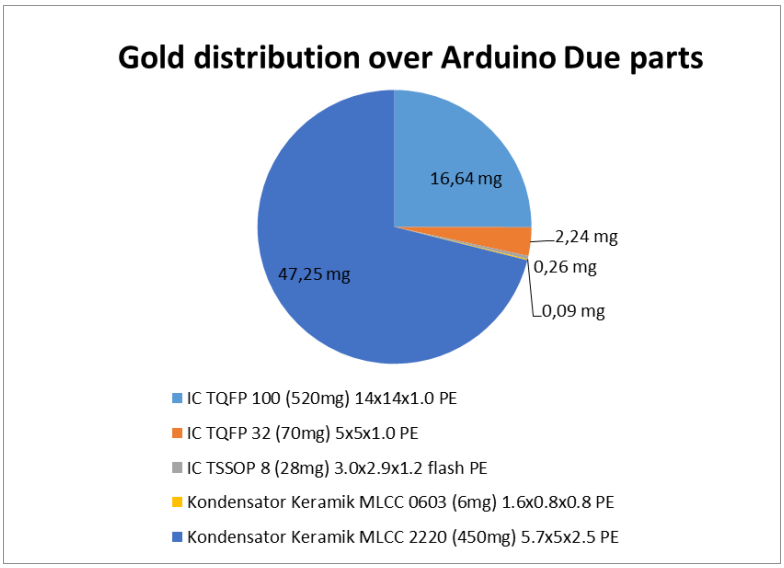


Figure 54: Gold distribution over Arduino Due parts

Another valuable and recyclable precious metal is palladium. The palladium distribution over the parts of the Arduino Due is shown in Figure 55.

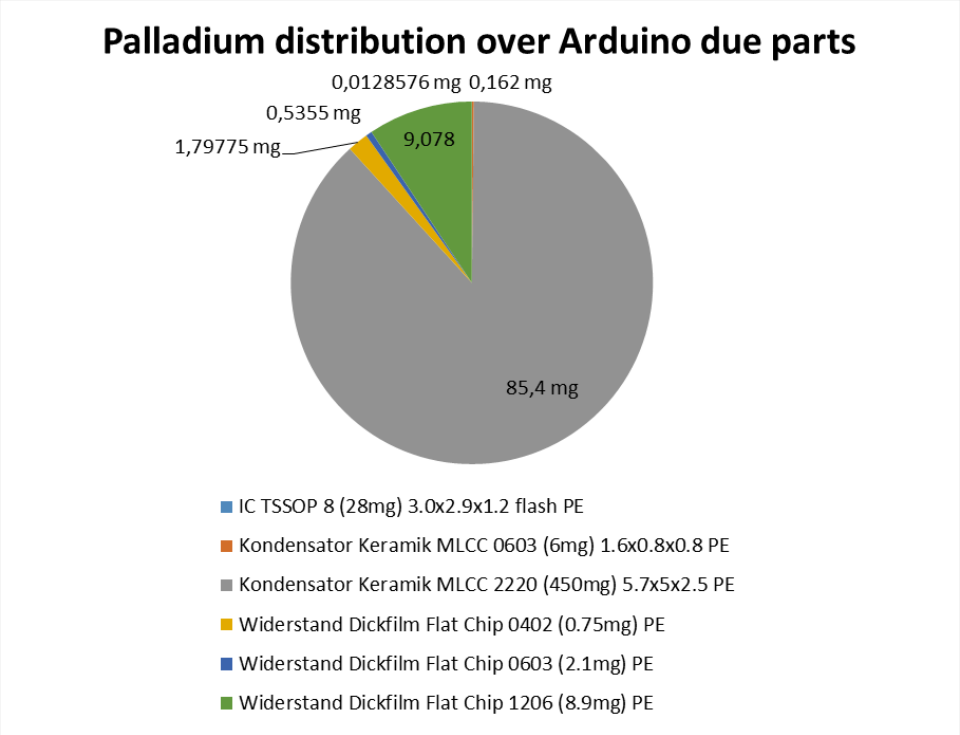


Figure 55: Palladium distribution over Arduino Due parts

Electronic components can be unsoldered from PCBs and reused in other electronic applications. Due to the high price fluctuation between electronic components and the high cost of unsoldering and testing electronic parts for reuse, the component prices are a strong indicator for reusability. The prices of the Arduino Due part with a price greater 0.1€ are shown in Figure 56 the price of the rest of the parts are summarized to Rest.

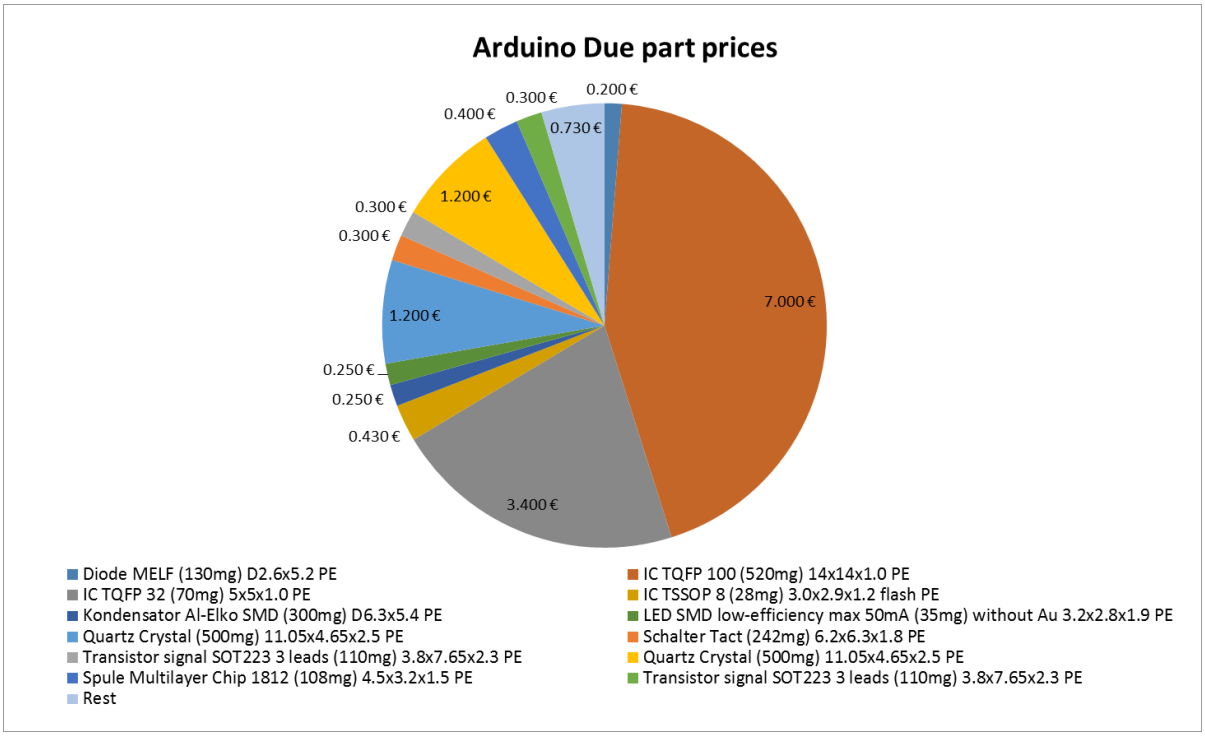


Figure 56: Arduino Due part prices

- What is arduino Due board
- - 17 No, 16 Yes
- - palladium MLCC article

6. Discussion and future work

Sad

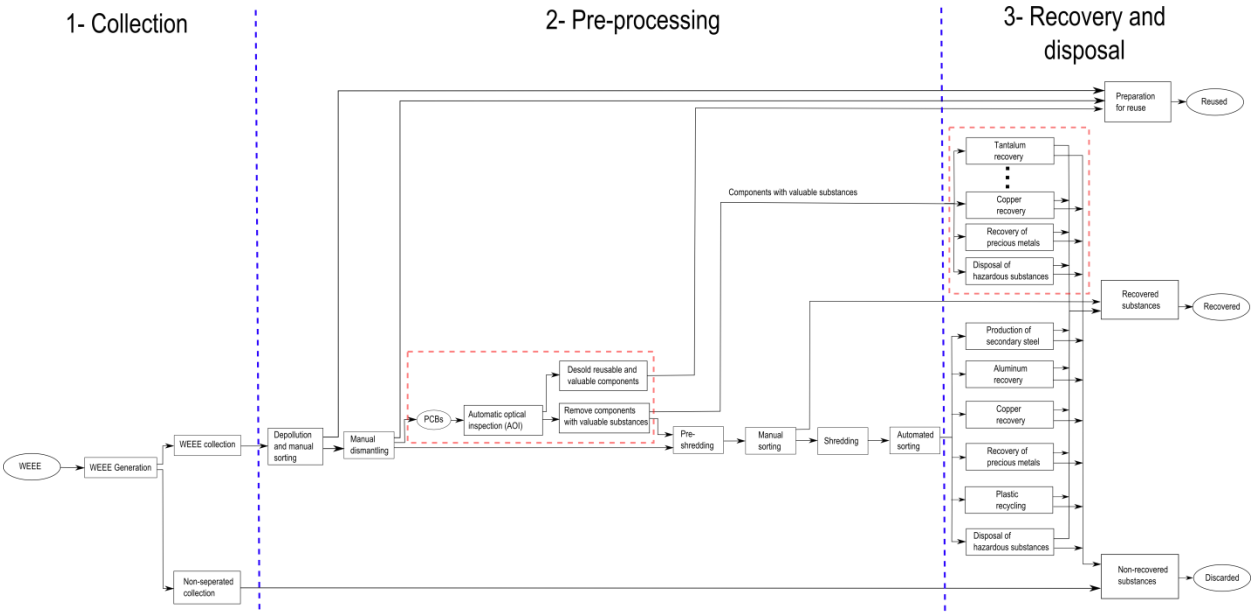
- Lasertrangulation
- 3D model

7. Conclusion

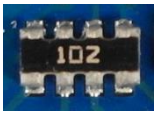
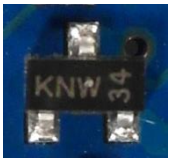
est

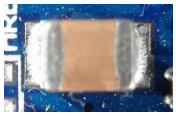
7.1 Application inclusion in a PCB recycling process chain

Dsf



Appendix A Recognition database components

Component name and description	Component image
Tantalum capacitor - Package: EIA Code 2412 - Color: yellow/orange - Tantalum and aluminum electrolytic capacitor with solid electrolyte polarity markings	
SMD Aluminum electrolytic capacitor - Diameter: 6.5 mm	
QFP100 - Package: QFP100 - 23.4 mm x 17.4 mm	
SMD Resistor Network array 1206, 4 Resistors - Long Side Terminals - Four resistors	
SMD Transistor SOT23-3 - Package: SOT23-3 - 3.0 mm x 2.6 mm	
DIP14 - Package: DIP14 - 19.5 mm x 7.6 mm	

DIP16 - Package: DIP14 - 19.5 mm x 7.6 mm	
SMD Resistor 1206 - Imperial code: 1206 (3216 metric) - 3.2 mm x 1.6 mm	
SOIC-8 - Package: SOIC8 - 5.0 mm x 6.2 mm	
Ceramic capacitor 1210 - Imperial code: 1210 (3225 metric) - 3.2 mm x 2.5 mm - Color: brown/orange	
SOT223-3 - Package: SOT223-3 - 6.5 mm x 7.0 mm	
SMD Resistor 0806 - Imperial code: 0806 (2012 metric) - 2.0 mm x 1.2 mm	

TO 263 - Imperial code: 0806 (2012 metric) - 10.1 mm x 15.0 mm	
Quartz HC-49/S - Package: HC-49/S-3 - 4.7 mm x 11.0 mm	
PCI - 32-bit PCI slot - 9.0 mm x 85.0 mm	

Appendix B Most important selected features

	Most important Frequency features	Most important Color features	Most important Segment features	Most important PCA reconstruction features	Most important Features selection from all feature sets
Tantalum capacitor	34, 45,				
SMD Aluminum electrolytic capacitor					
QFP100					
SMD Resistor Network array 1206, 4 Resistors					
SMD Transistor SOT23-3					
DIP14					
DIP16					
SMD Resistor 1206					
SOIC-8					
Ceramic capacitor 1210					
SOT223-3					
SMD Resistor 0806					

Appendix C Random forest classification results

		Frequency features	Color features	Segment features	PCA reconstruction feature	Features selection from all feature sets
Tantalum capacitor	True positive	59/59 (100%)	59/59 (100%)	52/52 (100%)	47/52 (90.4%)	59/59 (100%)
	False positive	59/59 (100%)	59/59 (100%)	47/48 (97.9%)	41/48 (85.4%)	59/59 (100%)
SMD Aluminum electrolytic capacitor	True positive	107/112 (95.5%)	107/112 (95.5%)	88/94 (93.6%)	88/94 (93.6%)	110/112 (98.2)
	False positive	111/112 (99.1%)	106/112 (94.6%)	83/96 (86.5%)	90/96 (93.8%)	111/112 (99.1%)
QFP100	True positive	75/79 (94.9%)	76/79 (96.2%)	62/65 (95.4%)	59/65 (90.8%)	79/79 (100%)
	False positive	77/79 (97.5%)	75/79 (94.9%)	65/69 (94.2%)	59/69 (85.5%)	77/79 (97.5%)
SMD Resistor Network array 1206, 4 Resistors	True positive	264/266 (99.2%)	260/266 (97.7%)	211/225 (93.8%)	219/225 (97.3%)	265/266 (99.6%)
	False positive	266/266 (100%)	255/266 (95.9%)	204/227 (89.9%)	223/227 (98.2%)	265/266 (99.6%)
SMD Transistor SOT23-3	True positive	125/126 (99.2%)	126/126 (100%	99/105 (94.3%)	105/105 (100%)	126/126 (100%)
	False positive	137/137 (100%)	127/137 (92.7%)	110/117 (94.0%)	1167117 (99.1%)	137/137 (100%)
DIP14	True positive	111/114 (97.4%)	104/114 (91.2%)	91/99 (91.9%)	95/99 (96.0%)	112/114 (98.2%)
	False positive	112/114 (98.2%)	105/116 (92.1%)	85/95 (89.5%)	91/95 (95.8%)	112/114 (98.2%)
DIP16	True positive	69/72 (95.8%)	67/72 (93.0%)	51/57 (89.5%)	50/57 (87.7%)	71/72 (98.6%)
	False positive	70/72 (97.2%)	67/72 (93.0%)	59/65 (90.8%)	52/65 (80.0%)	72/72 (100%)
SMD Resistor 1206	True positive	260/266 (97.7%)	260/266 (97.7%)	215/226 (95.1%	222/226 (98.2%)	261/266 (98.1%)
	False positive	266/266 (100%)	258/266 (97.0%)	208/226 (92.0%)	220/226 (97.3%)	266/266 (100%)
SOIC-8	True positive	104/106 (98.1%)	105/106 (99.1%)	83/87 (94.3%)	87/88 (98.9%)	104/106 (98.1%)

	False positive	105/106 (99.0%)	96/106 (90.6%)	76/92 (82.6%)	92/92 (100%)	106/106 (100%)
Ceramic capacitor 1210	True positive	36/42 (85.7%)	41/42 (97.6%)	35/36 (97.2%)	24/36 (66.7%)	41/42 (97.6%)
	False positive	34/42 (81.0%)	38/42 (90.5%)	34/35 (97.1%)	24/65 (68.6%)	41/42 (97.6%)
SOT223-3	True positive	257/262 (98.1%)	257/262 (98.1%)	212/222 (95.5%)	205/222 (92.3%)	262/262 (100%)
	False positive	258/262 (98.5%)	250/262 (95.4%)	201/223 (90.1%)	207/223 (92.8%)	262/262 (100%)
SMD Resistor 0806	True positive	294/308 (95.4%)	302/308 (98.1%)	249/258 (96.5%)	242/258 (93.8%)	306/308 (99.4%)
	False positive	306/308 (99.4%)	290/308 (94.2%)	252/266 (94.7%)	247/268 (92.9%)	303/308 (98.4%)
TO263	True positive	36/36 (100%)	32/26 (88.9%)	26/29 (89.7%)	28/29 (96.6%)	34/36 (94.4%)
	False positive	36/36 (100%)	30/36 (83.3%)	29/32 (90.6%)	31/32 (96.8%)	35/36 (97.2%)
Quartz HC-49/S	True positive	36/46 (93.5%)	46/46 (100%)	31/37 (83.8%)	27/37 (73.0%)	46/46 (100%)
	False positive	43/46 (93.5%)	45/46 (97.8%)	35/36 (97.2%)	29/36 (80.6%)	45/46 (97.8%)
32-bit-PCI slot	True positive	76/77 (98.7%)	76/77 (98.7%)	60/63 (95.2%)	62/63 (98.4%)	77/77 (100%)
	False positive	76/77 (98.7%)	71/77 (92.2%)	63/68 (92.6%)	67/68 (98.5%)	76/77 (98.7%)

Appendix D Linear-SVM classification results

		Frequency features	Color features	Segment features	PCA reconstruction feature	Features selection from all feature sets
Tantalum capacitor	True positive	59/59 (100%)	58/59 (98.3%)	52/52 (100%)	45/52 (86.5%)	59/59 (100%)
	True positive	57/59 (96.6%)	58/59 (98.3%)	43/47 (89.6%)	45/48 (93.8%)	59/59 (100%)
SMD Aluminum electrolytic capacitor	True positive	108/112 (96.4%)	108/112 (96.4%)	92/94 (97.9%)	87/94 (96.9%)	110/112 (98.2%)
	False positive	109/112 (97.3%)	101/112 (90.2%)	70/96 (72.9%)	93/96 (96.9%)	112/112 (100%)
QFP100	True positive	78/79 (98.7%)	77/79 (97.5%)	61/65 (93.8%)	57/65 (87.7%)	79/79 (100%)
	False positive	75/79 (94.9%)	75/79 (94.9%)	62/69 (89.9%)	67/69 (97.1%)	79/79 (100%)
SMD Resistor Network array 1206, 4 Resistors	True positive	261/266 (98.1%)	258/266 (97.0%)	225/255 (100%)	222/225 (98.7%)	265/266 (99.6%)
	False positive	265/266 (99.6%)	231/266 (86.8%)	188/227 (82.8%)	224/227 (98.7%)	264/266 (99.2%)
SMD Transistor SOT23-3	True positive	258/262 (98.5%)	255/262 (97.3%)	217/223 (97.7%)	207/223 (93.2%)	261/262 (99.6%)
	False positive	259/262 (98.9%)	239/262 (91.2%)	179/223 (80.3%)	215/224 (96.4%)	258/262 (98.5%)
DIP14	True positive	111/114 (97.4%)	104/114 (91.2%)	93/99 (93.9%)	95/99 (96.0%)	112/114 (98.2%)
	False positive	109/114 (95.6%)	98/114 (86.0%)	88/95 (92.6%)	93/95 (97.9%)	113/114 (99.1%)
DIP16	True positive	65/72 (90.3%)	69/72 (95.8%)	53/57 (93.0%)	47/57 (82.5%)	71/72 (98.6%)
	False positive	70/72 (97.2%)	63/72 (87.5%)	56/65 (86.2%)	61/65 (93.8%)	71/72 (98.6%)
SMD Resistor 1206	True positive	264/266 (99.2%)	256/266 (96.2%)	218/226 (96.5%)	219/226 (96.9%)	265/266 (99.6%)
	False positive	262/266 (98.5%)	237/266 (89.1%)	192/226 (85.0%)	223/226 (98.7%)	265/266 (99.6%)
SOIC-8	True positive	103/106 (97.2%)	102/106 (96.2%)	82/88 (93.2%)	86/88 (97.7%)	103/106 (97.2%)

		101/106 (95.3%)	93/106 (87.7%)	78/92 (84.8%)	92/92 (100%)	104/106 (98.1%)
Ceramic capacitor 1210	True positive	34/42 (81.0%)	42/42 (100%)	35/36 (97.2%)	29/38 (80.6%)	42/42 (100%)
	False positive	29/42 (69.0%	39/42 (92.9%)	28/35 (80.0%)	25/35 (71.4%)	39/42 (92.9%)
SOT223-3	True positive	126/126 (100%)	116/126 (92.1%)	99/105 (94.3%)	105/105 (100%)	126/126 (100%)
	False positive	137/137 (100%)	107/137 (78.1%)	84/117 (71.8%)	116/117 (99.1%)	137/137 (100%)
SMD Resistor 0806	True positive	289/308 (93.8%)	296/308 (96.1%)	257/258 (99.6%)	241/258 (93.4%)	308/308 (100%)
	False positive	285/308 (92.5%)	276/308 (89.6%)	230/266 (86.6%)	255/266 (95.9%)	299/308 (97.1%)
TO263	True positive	35/36 (97.2%)	31/36 (86.1%)	25/29 (86.2%)	28/29 (96.6%)	34/36 (94.4%)
	False positive	36/36 (100%)	30/36 (83.3%)	28/32 (87.5%)	32/32 (100%)	36/36 (100%)
Quartz HC-49/S	True positive	43/46 (93.5%)	44/46 (95.7%)	32/27 (86.5%)	28/37 (75.7%)	46/46 (100%)
	False positive	45/46 (97.8%)	42/46 (91.3%)	33/36 (91.7%)	34/36 (94.4%)	43/46 (93.5%)
32-bit-PCI slot	True positive	77/77 (100%)	77/77 (100%)	60/63 (95.2%)	63/63 (100%)	77/77 (100%)
	False positive	76/7 (98.7%)	71/77 (92.2%)	63/68 (92.6%)	66/68 (97.1%)	76/77 (98.7%)

Appendix E RBF-SVM classification results

		Frequency features	Color features	Segment features	PCA reconstruction feature	Features selection from all features sets
Tantalum capacitor	True positive	59/59 (100%)	59/59 (100%)	52/52 (100%)	47/52 (90.4%)	59/59 (100%)
	True positive	59/59 (100%)	59/59 (100%)	47/48 (97.9%)	41/48 (85.4%)	59/59 (100%)
SMD Aluminum electrolytic capacitor	True positive	107/112 (95.5%)	107/112 (95.5%)	88/94 (93.6%)	88/94 (93.6%)	110/112 (98.2%)
	False positive	111/112 (99.1%)	106/112 (94.6%)	83/96 (86.5%)	90/96 (93.8%)	111/112 (99.1%)
QFP100	True positive	75/79 (94.9%)	76/79 (96.2%)	62/65 (95.4%)	59/65 (90.8%)	79/79 (100%)
	False positive	77/79 (97.5%)	75/79 (94.9%)	65/69 (94.2%)	59/69 (85.5%)	77/79 (97.5%)
SMD Resistor Network array 1206, 4 Resistors	True positive	264/266 (99.2%)	260/266 (97.7%)	211/225 (93.8%)	219/225 (97.3%)	265/266 (99.6%)
	False positive	266/266 8100%)	255/266 (95.9%)	204/227 (89.9%)	223/227 (98.2%)	265/266 (99.6%)
SMD Transistor SOT23-3	True positive	257/262 (98.1%)	257/262 (98.1%)	212/222 (95.5%)	205/222 (92.3%)	262/262 (100%)
	False positive	258/262 (98.5%)	250/262 (95.4%)	201/223 (90.1%)	207/223 (92.8%)	262/262 (100%)
DIP14	True positive	111/114 (97.4%)	104/114 (91.1%)	91/99 (91.9%)	95/99 (96.0%)	112/114 (98.2%)
	False positive	112/114 (98.2%)	105/114 (89.5%)	85/95 (89.5%)	91/95 (95.8%)	112/114 (98.2%)
DIP16	True positive	69/72 (95.8%)	67/72 (93.0%)	51/57 (89.5)	50/57 (87.7%)	71/72 (98.6%)
	False positive	70/72 (97.2%)	67/72 (93.0%)	59/65 (90.8%)	52/65 (80.0%)	72/72 (100%)
SMD Resistor 1206	True positive	260/266 (97.7%)	260/266 (97.7%)	215/226 (95.1%)	222/226 (98.2%)	261/266 (98.1%)
	False positive	266/266 (100%)	258/266 (97.0%)	208/226 (92.0%)	220/226 (97.3%)	266/266 (100%)
SOIC-8	True positive	104/106 (98.1%)	105/106 (99.1%)	83/88 (94.3%)	87/88 (98.9%)	104/106 (98.1%)

	False positive	105/106 (99.1%)	96/106 (90.6%)	76/92 (82.6%)	92/92 (100%)	106/106 (100%)
Ceramic capacitor 1210	True positive	36/42 (85.7%)	41/42 (97.6%)	35/36 (97.2%)	24/36 (66.6%)	41/42 (97.6%)
	False positive	34/42 (81.0%)	38/42 (90.5%)	34/35 (97.1%)	24/35 (68.6%)	41/42 (97.6%)
SOT223-3	True positive	125/126 (99.2%)	126/126 (100%)	99/105 (94.3%)	105/105 (100%)	126/126 (100%)
	False positive	137/137 (100%)	127/137 (92.7%)	110/117 (94.0%)	116/117 (99.1%)	137/137 (100%)
SMD Resistor 0806	True positive	294/308 (95.4%)	302/308 (98.0)	249/258 (96.5%)	242/258 (93.8%)	306/308 (99.4%)
	False positive	306/308 (99.3%)	290/308 (94.2%)	252/266 (94.7%)	247/266 (92.9%)	303/308 (98.4%)
TO263	True positive	36/36 (100%)	32/36 (88.9%)	26/29 (88.9%)	28/29 (96.6%)	34/36 (94.4%)
	False positive	36/36 (100%)	30/36 (83.3)	29/32 (90.6%)	31/32 (96.9%)	35/36 (97.2%)
Quartz HC-49/S	True positive	36/46 (78.3%)	46/46 (100%)	31/37 (83.8%)	27/37 (73.0%)	46/46 (100%)
	False positive	43/46 (93.5%)	45/46 (97.8%)	35/36 (97.2%)	29/36 (80.6%)	45/46 (97.8%)
32-bit-PCI slot	True positive	76/77 (98.7%)	76/77 (98.7%)	60/63 (95.2%)	62/63 (98.4%)	77/77 (100%)
	False positive	76/77 (98.7%)	71/77 (92.2%)	63/68 (92.6%)	67/68 (98.5%)	76/77 (98.7%)

Appendix F Basis weight determination (PCB mounted)

Length $[cm]$	Width $[cm]$	Weight $[g]$	Area $[cm^2]$	Basis weight $[\frac{g}{cm^2}]$
26	23	450	598	0.752508361
17	5.5	110	93.5	1.176470588
31	24	670	744	0.900537634
14	19	110	266	0.413533835
23	10	160	230	0.695652174
19	14	110	266	0.413533835
11	25	170	275	0.618181818
31	24	620	744	0.833333333
24	24	400	576	0.694444444
24	16	250	384	0.651041667
20	14	145	280	0.517857143
24	19	440	456	0.964912281
19	14	200	266	0.751879699
27	15	275	405	0.679012346
17	8.5	120	144.5	0.830449827
13	10	90	130	0.692307692
30.5	22	600	671	0.894187779
16	16	150	256	0.5859375
8.5	5.5	35	46.75	0.748663102
14	5.5	70	77	0.909090909
12	7	70	84	0.833333333
19	14	105	266	0.394736842
18	10	150	180	0.833333333
17	10	200	170	1.176470588
		5700	**7608.75**	**0.749137506**

Appendix G Arduino Due component replacement model

Arduino Due component package	GaBi component replacement model	Large component deviation
SMC_B	Kondensator Keramik MLCC 2220 (450mg) 5.7x5x2.5	No
C0402	Kondensator Keramik MLCC 01005 (0,054 mg) 0,4x0,2x0,22	No
C0603	Kondensator Keramik MLCC 0603 (6mg) 1.6x0.8x0.8 (Nichtedelmetalle)	No
SMB	Transistor signal SOT23 3 leads (10mg) 1.4x3x1	Yes
MINIMELF	Diode MELF (130mg) D2.6x5.2	No
DO220AAL	Diode power DO214_219 (93mg) 4.3x3.6x2.3	No
SMD_1575SW	Schalter Tact (242mg) 6.2x6.3x1.8	Yes
L1812	Spule Multilayer Chip 1812 (108mg) 4.5x3.2x1.5	No
MSOP08	IC TSSOP 8 (28mg) 3.0x2.9x1.2 flash	Yes
SOT23-6	Transistor signal SOT23 8 leads (18mg) 1.4x3x2	Yes
SOT223	Transistor signal SOT223 3 leads (110mg) 3.8x7.65x2.3	No
MLF32	IC TQFP 32 (70mg) 5x5x1.0	Yes
SC70-5	IC TSSOP 8 (28mg) 3.0x2.9x1.2 DRAM	Yes
R0402	Widerstand Dickfilm Flat Chip 0402 (0.75mg)	No
CHIP-LED0805	LED SMD low-efficiency max 50mA (35mg) without Au 3.2x2.8x1.9	No
L1 0805	Spule Multilayer Chip 1812 (108mg) 4.5x3.2x1.5	Yes
SRR0604	Spule Miniatur gewickelt SDR1006 (1.16g) D9.8x5.8	No
PANASONIC_D	Kondensator Al-Elko SMD (300mg) D6.3x5.4	No
SOT23	Transistor signal SOT23 3 leads (10mg) 1.4x3x1	No
R0603	Widerstand Dickfilm Flat Chip 0603 (2.1mg)	No
TS42	Schalter Tact (242mg) 6.2x6.3x1.8	Yes
CAT16	4 x Widerstand Dickfilm Flat Chip 1206 (8.9mg)	Yes
SOT-23	Transistor signal SOT23 3 leads (10mg)	No

	1.4x3x1	
LQFP144	IC TQFP 100 (520mg) 14x14x1.0	Yes
CRYSTAL-3.2-2.5	0.5 x Quartz Crystal (500mg) 11.05x4.65x2.5	Yes
RESONATOR_EPSON_FC_145	1 x Quartz Crystal (500mg) 11.05x4.65x2.5	Yes
CT/CN0603	Widerstand Dickfilm Flat Chip 0603 (2.1mg)	No
PINHD-2x3	1.5 x Widerstand Dickfilm Flat Chip 0402 (0.75mg)	Yes
J0402	Kondensator Keramik MLCC 01005 (0,054 mg) 0,4x0,2x0,22	Yes
MCR-AB1-S-RA-SMT	Stecker, für Netzwerkkabel, ab Werk	Yes
POWERSUPPLY_DC-21MM	Stecker, für Netzwerkkabel, ab Werk	Yes
FR4 glass epoxy	Leiterplatte 2-Lagen starr FR4	No
Solder SnAg3.5	Lotpaste SnAg	No

Appendix H Metal prices

Metal	Price [$/kg]	Data source
Aluminium (E) [kg]	2.1	www.lme.com, 27.11.2014
Aluminiumoxid (Al2O3) (E) [kg]	0	-
Anorganische Flammschutzmittel (E) [kg]	0	-
Antimonoxid (Diantimontrioxid) (Sb2O3) (E) [kg]	0	-
Blei (E) [kg]	2.0	www.lme.com, 27.11.2014
Blei in Legierung (E) [kg]	2.0	www.lme.com, 27.11.2014
Chrom in Legierung (E) [kg]	2.7	www.metal.page.com, 27.11.2014
Cobalt in Legierung (E) [kg]	31.0	www.lme.com, 27.11.2014
Eisen in Legierung (E) [kg]	0	-
Elektrolyt (E) [kg]	0	-
Epoxidharz (EP) (E) [kg]	0	-
Ethylen-Propylen-Dien-Kautschuk (EPDM) (E) [kg]	0	-
Ferrite (E) [kg]	0	-
Glas (E) [kg]	0	-
Glasfasern (E) [kg]	0	-
Gold (E) [kg]	42400	www.infomin.com, 27.11.2014
Gold in Legierung (E) [kg]	42400	www.infomin.com, 27.11.2014
Kupfer (E) [kg]	6.6	www.lme.com, 27.11.2014
Kupfer in Legierung (E) [kg]	6.6	www.lme.com, 27.11.2014
Nickel (E) [kg]	16.3	www.lme.com, 27.11.2014
Nickel in Legierung (E) [kg]	16.3	www.lme.com, 27.11.2014
Palladium in Legierung (E) [kg]	29000	www.infomin.com, 27.11.2014
Papier (E) [kg]	0	-
Platin in Legierung (E) [kg]	42800	www.infomin.com, 27.11.2014
Polyamid 6 (PA6) (E) [kg]	0	-
Polyphenylensulfid (PPS) (E) [kg]	0	-
Polystyrol (PS) (E) [kg]	0	-
Polytetrafluorethylen (PTFE) (E) [kg]	0	-
Silber (E) [kg]	592.0	www.infomin.com, 27.11.2014
Silber in Legierung (E) [kg]	592.0	www.infomin.com, 27.11.2014
Silicium (E) [kg]	0	-
Siliciumdioxid (SiO2) (E) [kg]	0	-
Silikon (Sl) (E) [kg]	0	-
Stahl, unlegiert (Fe-C) (E) [kg]	0.5	www.lme.com, 27.11.2014
Tetrabrombisphenol A (TBBA) (E) [kg]	0	-

Ungesättigte Polyester (UP) (E) [kg]	0	-
Zink (E) [kg]	2.25	www.lme.com, 27.11.2014
Zink in Legierung (E) [kg]	2.25	www.lme.com, 27.11.2014
Zinn (E) [kg]	20.3	www.lme.com, 27.11.2014
Zinn in Legierung (E) [kg]	20.3	www.lme.com, 27.11.2014