An Efficient Gradient based Registration Technique for Coin Recognition

Marco Reisert, Olaf Ronneberger, Hans Burkhardt

Albert-Ludwig University

Georges Koehler Allee 52

Tel. 0761/203-8267

e-mail: reisert@informatik.uni-freiburg.de

Abstract

This paper presents a coin recognition system based completely on the direction of the gradient vectors. To optimally align two coins we search for a rotation such that as most as possible corresponding gradient vectors point into the same direction. After discretizing the gradient directions this can be done quickly by the use of the Fast Fourier Transform. The classification is done by a simple nearest neighbor search followed by several rejection criteria to meet the demand of a low false positive rate.

1 Introduction

The goal of a coin recognition system is to automatically sort and classify high volumes of coins with high accuracy within a small amount of time. Usually the classification is based on images from both sides of the coin, its radius and thickness. With the changeover of 12 European currencies to the Euro this task has gained much more interest. In 2003 ARC Seibersdorf research GmbH created the sorting device called *Dagobert* [1]. The recognition unit of Dagobert is able to discriminate between over 600 different coin types based on over 2000 different coin faces. This work was developed in conjunction with the *Coin Image Seibersdorf (CIS) Benchmark*, [1] which was defined to foster the development of robust recognition and image search algorithms.

This work roughly follows the ideas proposed in [1]. The similarity of two coin images is computed by the use of registration techniques. In a first step the translational pose of the coin is determined by some segmentation algorithm that makes an estimate of the coin's radius and its center. The comparison of two coins is done by aligning them with respect to their rotational pose. Having defined a similarity measure any classification scheme may be used. Instead of registration techniques one might argue to use methods based on rotational invariant features. But due to the requirements of very low false positive rates, we believe that registration is unavoidable to obtain reliable rejection criteria. Another reason for the registration approach

[1] http://muscle.prip.tuwien.ac.at/index.php

is that there are, in fact, accurate correspondences that should be used for recognition. If two coins are stemming from the same class they are, in their original appearance, exactly identical, so a registration is predestined for recognition. The only difficulty is to find robust similarity measures that tolerate the, sometimes severe, abrasion and fouling of the coins, but still give response for fine structures determining the class membership. Additionally this similarity measure has to be computable in a fast manner such that the resulting algorithm can cope with large databases.

This work is organized as follows: In the following subsection we give a short overview of other coin recognition systems. In Section 2 we present our algorithm for coin segmentation that is based on the Hough Transform. In Section 3 we present the features that are used for alignment and similarity computation and give further some implementation details. The classification scheme is presented in Section 4. Finally we show some exemplary experiments. In Section 6 we give a conclusion and ideas for further improvement.

1.1 Previous Work

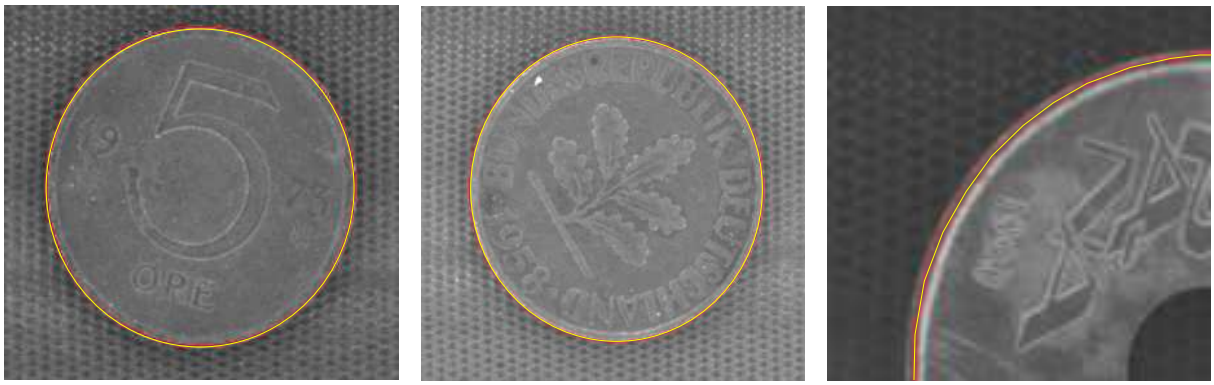
Recent approaches for coin recognition can roughly be divided in methods based on rotational invariant features and methods based on registration. In [5, 6] invariant features are used to compare coins in a rotational invariant manner. We already claimed, such methods are not as well suited for our purposes. For examples in [5] very high false positive rates are reported. In [1, 7] the similarity computation is based on registration techniques. It seems possible that these approaches are able to fulfill the reliability demanded in the benchmark specifications.

2 Coin Segmentation

In [1] a simple segmentation by thresholding the grayvalues is used. This was based on the assumption that the coins itself are brighter than its background. Having a look at the benchmark database this is mostly but not always the case. So, we have to search for another solution. The Hough Transform is known to be a very robust segmentation Tool. We use a kind of Generalized Hough Transform (GHT) [2] to segment the coins. Recently, this type of GHT was used for fast segmentation of cell-nuclei [3]. We use a three dimensional voting space, the two coordinates of the coin's center and its radius r. Let us call the gray-valued image function $I : \mathbb{R}^2 \mapsto \mathbb{R}$ and the voting function $P : \mathbb{R}^3 \mapsto \mathbb{R}$. The idea is to search for maxima of the following integral

$$P(\mathbf{x}, r) = \int_{\mathbb{R}^2} \left(\delta(\mathbf{x} + r\frac{\nabla I(\mathbf{y})}{||\nabla I(\mathbf{y})||}) + \delta(\mathbf{x} - r\frac{\nabla I(\mathbf{y})}{||\nabla I(\mathbf{y})||}) \right) \, ||\nabla I(\mathbf{y})|| \, d\mathbf{y},$$

where δ is some indicator function giving contribution whenever its argument is nearby zero, for example a gaussian or a rectangle-function. Each gradient in the image votes for a possible center of the coin, where the hypothetical center has to be along the gradient's direction. Since we do not know whether the coin is darker or brighter than the background we have to vote in positive and negative direction.



a) b) c)

Figure 1: Coins segmented by our algorithm. Coin a) and b) have very bad contrast conditions, however our algorithm is able to make a good segmentation. In c) a section of a coin is shown where our algorithm makes a small mistake. Due to a strong circle-like structure at the border of the coin the estimated radius is a little bit to small. However, this is a systematical error, and if this mistake is done for every coin from the specific class it should not confuse the classification algorithm.

Before starting to compute the integral we blur the image with a first-order IIR-Filter to get more well-behaved gradients. The computation of the integral is straightforward. Just run linearly over the image I and compute the gradient $\nabla I(\mathbf{y})$ and its magnitude. Now for every position $\mathbf{y}$ and for discrete radii values r_i accumulate the voting function at positions $\mathbf{y} + r_i \frac{\nabla I(\mathbf{y})}{||\nabla I(\mathbf{y})||}$ and $\mathbf{y} - r_i \frac{\nabla I(\mathbf{y})}{||\nabla I(\mathbf{y})||}$ with weight $||\nabla I(\mathbf{y})||$. The shape of the accumulation depends on the indicator function δ. We just round the estimated coordinates to the nearest integers and accumulate the single pixels. After all accumulations are done the voting function P is smoothed by an IIR-Filter. So the indicator function is just the impulse response of this filter.

Since we know that only one coin is present in the image, we can use a hierarchical voting for the radius to get better estimates. For a first rough estimate we take 16 different values for the radius covering the whole range of possible coin sizes. After the determination of the first maximum we distribute again 16 radius bins around the first rough estimate. Overall we repeat this procedure four times resulting in an accurate estimate. On an *Intel P4 2.8Ghz* the overall segmentation procedure needs less than one second for one image. In Figure 1 we show three examples of segmented coins.

3 Feature Extraction and Registration

Having a look at the benchmark database one can guess that the actual gray values of the images are not very expressive. In [1] a Canny Edge-Detector is used to compute more reliable features. This is, of course, a much better idea than working on the plain gray values, but it has still some disadvantages. At first, the results depend heavily on the choice of the parameters of the edge-detector. These parameters have to be chosen properly depending on illumination

Figure 2: Abrasion effects. Due to abrasion several slight gradients are introduced near the edges. Not only the edges contain structural information that may be considered, also putatively flat regions contain valuable structure. Our algorithm also uses this information for recognition.

conditions and the quality of image. And secondly, the orientation of the edges are completely neglected. We want to follow a different approach. We want to use solely the direction of the gradients in the image and totally neglect its magnitude. This has the advantage that we are totally independent of illumination and contrast changes. One can argue that only considering the direction is a very dirty approach, since we also compute gradient directions in homogeneous, flat regions where theoretically the gradient has to be zero and hence has no reliable direction. But, however, there are several reason to follow this way. At first, the direction in homogeneous regions can be assumed to be equidistributed with respect to its angle, so such regions give only constant bias to the similarity measure defined below. Further, we do not need any threshold to be adjusted, i.e. we never need to decide whether there is an edge or not. This advantage becomes important considering Figure 2. Due to abrasion coins show typical patterns near structural steps and edges. One can see several slight gradients near the edges. It would be ignorant to neglect this information. An algorithm, which makes only use of edges has problems to also incorporate this information.

During segmentation we determined the radius and center of the coin. From now on we assume that the center is shifted to the origin and the image is scaled such that the radius of the coin equals to 1. For convenience we represent the image I in polar coordinates $I(r, \varphi)$ with $r \in [0, 1]$ and $\varphi \in [0, 2\pi]$. The basis for our features is the normalized gradient image $\mathrm{g} = (g_r, g_\varphi)^T$ given by

$$g_r(r,\varphi) = \frac{\partial_r I}{\epsilon + \sqrt{(\partial_r I)^2 + (\partial_\varphi I)^2}}, \quad g_\varphi(r,\varphi) = \frac{\partial_\varphi I}{\epsilon + \sqrt{(\partial_r I)^2 + (\partial_\varphi I)^2}},$$

where ϵ is a small positive constant avoiding division by zero. We choose the radial and tangential derivatives because they do not change while rotating the coin. A rotation of the coin just shifts the φ coordinate of g cyclically. Based on the gradient image we compute the angle image $a \in [0, \pi]$ given by

$$a(r,\varphi) = \mathit{sign}(g_\varphi(r,\varphi))\ \mathit{arcsin}(g_r(r,\varphi))$$

describing the angle to the tangent at the circle. The sign modification leads to invariance against inverting the contrast. We found that it is not important whether an edge is falling or growing, because, e.g. by dirt or abrasion of the coin, the contrast conditions are sometimes inverted. So only one half of the circle is represented by this function, the other half is mapped by point reflection to the former. The function a is now used for comparing two coins. Let a and a' be two feature functions from two different coins, we define their correlation function by

$$c(\phi) = \int_0^{2\pi} \int_0^1 \delta(a(r,\varphi) - a'(r,\varphi-\phi))\, dr d\varphi,$$

where δ is again some indicator function deciding whether two angles are different or equal. It can be imagined as some peak-like function. So $c(\phi)$ counts how often the angles of the gradients of the two coins coincide at a specific relative angle ϕ. So the maximum value of c is defined as our similarity measure of two coins

$$k(a,a') = \max_{\phi\in[0,2\pi]} c(\phi) \tag{1}$$

It is well known that a computing correlation function can be done efficiently in the Fourier domain. But c is not an ordinary correlation; to get us into the position to apply the Fourier transform we first have to rewrite our correlation function. For any function g there exists the so called convolution square root $g^{1/2}$, which fulfills the following relation

$$g(x-y) = \int_{\mathbb{R}} g^{1/2}(x-z) g^{1/2}(z-y)\, dz.$$

Using this formula for our indicator function δ we can rewrite the correlation

$$c(\phi) = \int_0^{2\pi} \int_0^1 \int_{\mathbb{R}} \underbrace{\delta^{1/2}(a(r,\varphi)-z)}_{f(r,\phi,z)} \underbrace{\delta^{1/2}(z-a'(r,\varphi-\phi))}_{f'(r,\varphi-\phi,z)}\, dz dr d\varphi.$$

Indeed we introduced an additional integration but now our correlation function looks like an ordinary correlation of two functions f and f', which can be computed in the Fourier domain. Let us call $f(r,\phi,z)$ our final feature function and $\tilde{f}(r,k,z) = \int f(r,\varphi,z) e^{-ik\varphi} d\varphi$ its Fourier transform with respect to the angle parameter. The feature with respect the z-parameter can be imagined as a some kind of indicator function contributing whenever there is a gradient with the specific angle z in the original image.

Using the Fourier features the Fourier representation of $c(\phi)$ looks then

$$\tilde{c}(k) = \int_0^1 \int_{\mathbb{R}} \tilde{f}^*(r,k,z) \tilde{f}'(r,k,z)\, dr dz,$$

where $\cdot^*$ is the complex-conjugate. The last step is just to transform $\tilde{c}(k)$ back into the spatial domain and search for its maxima.

3.1 Implementation

Of course all steps from above has to be done in a discrete manner, so in the following we explain how this was realized.

Preprocessing and Gradient Computation Given the segmented image in cartesian coordinates we first applied a blur using the same IIR-filter used for segmentation. To get the polar gradient image we directly sample the gradient from the blurred image with 1024 steps in angular direction and 256 steps in radial direction using bilinear interpolation. We used a stepsize of 1 pixel in the original image for computing the finite differences for the gradient. In radial direction we only sample from 0 to 0.9, i.e. we leave out the outer 10 percent of coin, because it seems that it mostly contains useless information (see also [1]). Then we blurred the image again with the IIR-filter and downsampled it to a desired size, for the competition we use a size of 256×64. Finally we normalize the gradients and compute the angles according to the equations from above.

Discretizing the Angles and Final Feature Computation The gradient angle parameter z is discretized in M steps, i.e. we have a stepwidth of $\Delta z = \pi/M$. So the final feature image consists of M binary images of size 256×64. The binary image with number i contains entries whenever $\frac{a(r,\varphi)}{\Delta z}$ falls within the interval $[i, i+1]$ with $0 \leq i < M$. In Figure 3 we visualize the computation procedure. Having a closer look at the angle function $a(r, \varphi)$ one can see that also regions with relatively low gradient magnitude contain valuable information.

To become more robust against small gradient changes we additionally use some kind of 'inverse' bilinear interpolation (also called fuzzy histograms, see [4]) to generate the entries. For each entry the two nearest pixel get a contribution depending linearly on the distance to the pixel's center. We also conduct some smoothing in radial direction to be robust against small shifts in radial direction. Such radial shifts usually come from small errors during the segmentation process.

Finally the Fast Fourier Transform (FFT) of the feature function in φ direction can be precomputed to speed up the comparison later, i.e. we compute the FFT of the rows of the binary feature images on the right of Figure 3.

4 Classification

For classification we use a very simple nearest neighbor scheme. This is mainly due to complexity and memory consideration. For example, considering the feature parameters used for the competition, one feature is of size $256 \cdot 64 \cdot 6 \cdot$ sizeof(float) $=$ 392KByte. The training database consists of over two thousand averaged coin images, so we already need nearly one GigaByte storage only for the averaged images. For training we only use this two thousand average images. A whole scan of those using the introduced similarity measure would take about 6 seconds on a *Pentium P4 2.8Ghz* for just one side of the coin, while the benchmark specifications only allow about 5 seconds to classify one coin, which involves the classification

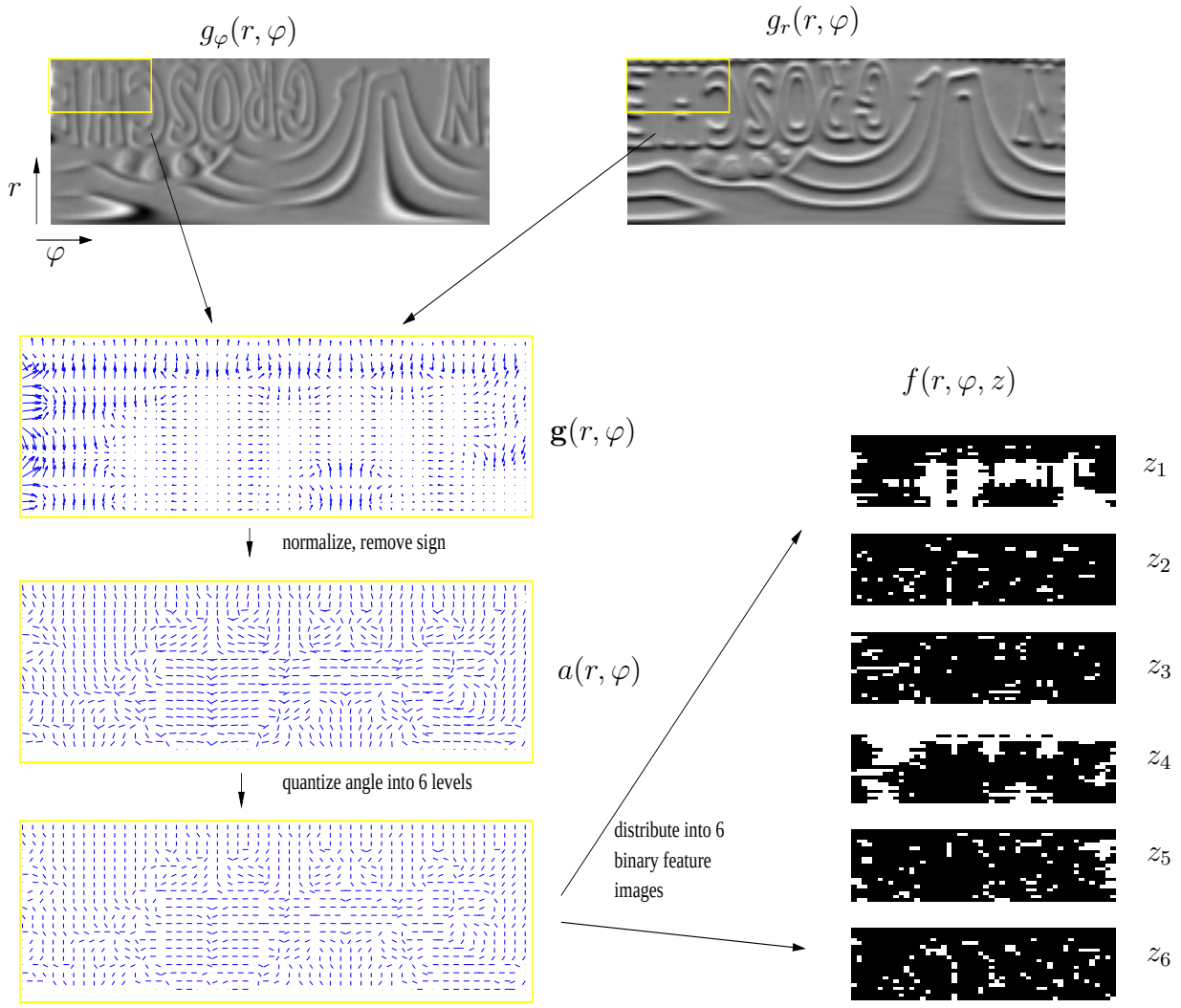


Figure 3: Sketch of feature extraction process. After the computation of radial and angular gradients the angle function $a(r,\varphi)$ is computed. For clarity we only visualize the yellow marked upper left corner of the image. In the $a(r,\varphi)$ the arrow-tips are omitted because the sign of the direction does not count anymore. One can see that also regions with relatively low gradient magnitude contain valuable, structured information. In the last step the directions are discretized in six discrete directions. Finally one binary feature image is created for each direction.

of both sides. To meet this goal we have to restrict us to search only in a subset of the database. Since we have a good estimate of the radius of the examined coin, the search is only performed in some neighborhood of this estimated radius. We search in a range of $2\Delta = 4mm$ around the estimated radius. Mostly, the search includes around 200 comparison depending on the coin's radius. In Figure 5 the distribution of the radius over all training examples is shown. After the algorithm determined the nearest neighbors within this range it is checked whether front and backside of the coin agree or do not agree. If they do not the coin is immediately rejected and classified as unknown. Otherwise we compute a prediction confidence C which is based on the similarity scores, thickness differences, radius differences and angle pose differences. First we normalize the similarity scores. Assuming that two images are compared that have totally random gradient orientation their similarity is expected to be $k = 256 \cdot 64/6 \approx 2730$, so we normalize our similarity score by $k' = k - 2730$. We further compute the thickness differences Δt_1 and Δt_2 for the front and backside of the coin to its matched partners from the training database. And we do the same for the radius. Due to the mechanics of the coin acquisition system front and backside of the coin should have a fixed rotational pose relation. We found that the relation is slightly unreliable, but we also included it with a small weight in our confidence score,

$$C = k_1' k_2' \exp(-\frac{(\Delta r_1 + \Delta r_2)}{0.17\ mm} - \frac{(\Delta t_1 + \Delta t_2)}{0.07\ mm} - \frac{(\Delta \phi)}{10^\circ}).$$

If this confidence is below 10^{-5} the coin is rejected. Additionally we make rejects if the thickness difference is above $0.25mm$ or the $\Delta\phi$ is above 70°. Besides, all this additional reject criteria do not have too much influence on the overall performance, the most powerful rejection criteria is definitely the consistency of the votes for the front and backside of the coin. In Figure 6 we give a rough overview over the complete classification system.

4.1 Special Cases

There are some classes that can confuse the proposed classification process. For example the coin types 0203 and 0202 have very similar backsides (see Figure 4). Hence there are often confusions between them and many coins of these types are rejected. To avoid this we introduced some kind of 'multitype' for the backside of this class. So, if the front side is e.g. 0203 then also 0202 is valid for the back side, but the final classification is still 0203. As a second problem we found that there are classes with very similar visual appearance but slightly different radius, e.g. 1321 and 1311 (see Figure 4). Here we also use the multitype scheme from above, and later decide depending on radius whether we have to vote for 1321 or 1311. The same problem arises for the classes 4027 and 4017. Finally we have a last special treatment for the classes 406 and 416, they have a small difference in its visual difference ('100' vs. 'Cien') but the overall appearance is very similar (see Figure 4). We found that our system had problems to discriminate between those, so we raise the confidence threshold from 10^{-5} to 10^2 if we detect type 406 or 416.

Figure 4: Example for three special cases. The right two images show the backside of classes 0203 and 0202 that are nearly identical. The images in the middle show instances of class 0406 and 0416 which are often confused. The right images show front and backside of class 1311 that has the same visual appearance as class 1321 but a different radius.

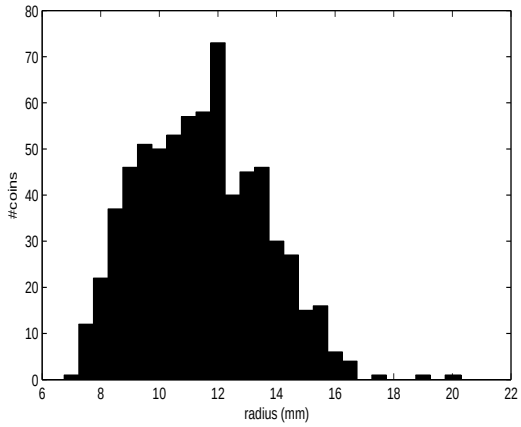


Figure 5: Distribution of the radius (mm) over the two thousand training images.

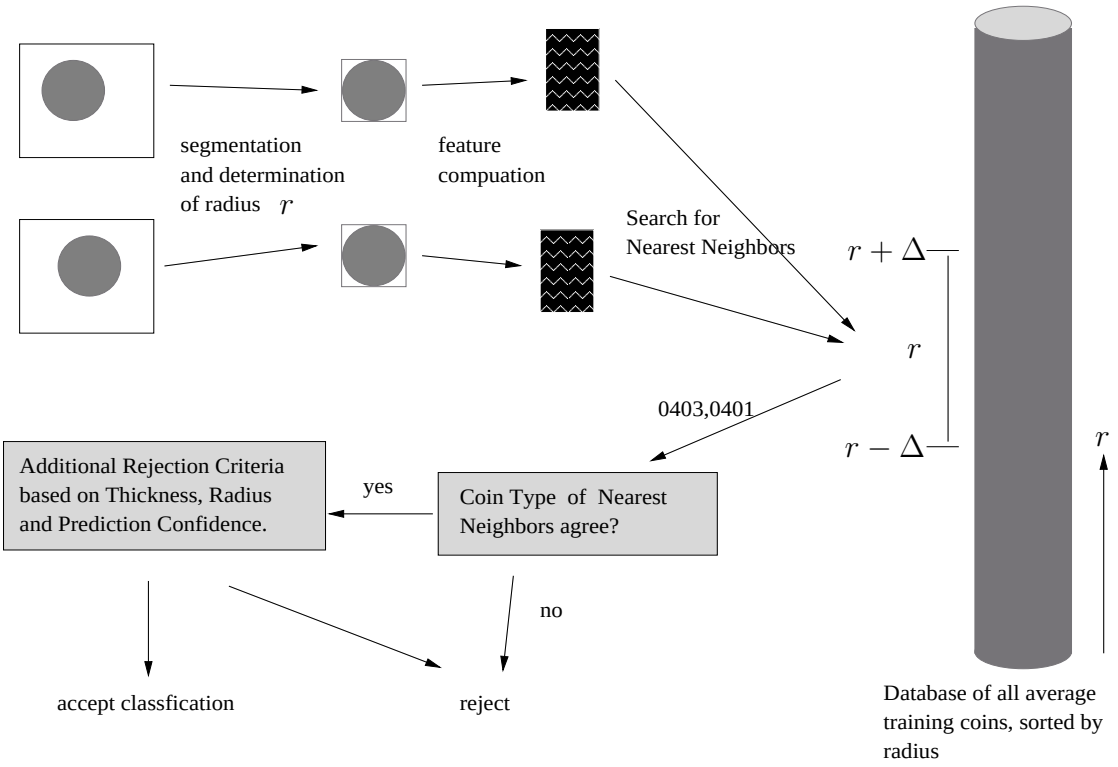


Figure 6: Complete workflow of the classification system. After segmentation of the coins the features $\tilde{f}(k, r, z)$ according to section 3 are computed. Then, nearest neighbors are searched with respect to the similarity measure (1). The search is only performed within a neighborhood of the estimated radius, not on the whole database. If the nearest neighbors of both coin sides agree the coin is a possible candidate for acceptance, otherwise it is immediately rejected. Finally additional rejection criteria are checked and some classes get a special treatment.

Tranche	1A (no rej.)	1A	1B	1C
RC	322	320	309	398
TP+TN (%)	97.64	97.27	97.20	94.31
FN+TN (%)	5.20	5.64	5.94	9.42
FP (%)	0.06	0.02	0.02	0.03
Ass. Score	17211	17527	17245	19081

Table 1: Experimental results for all three tranches. In the first column (no rej.) we give results without additional reject criteria. RC is the number of classes which got at least one correct classification. (T/F)(P/N) means True/False Positives/Negatives. TP+TN gives the total number of correct classification. FN+TN is the total number of rejects, i.e. the number of coins classified as unknown. FP gives the number of unknown coins, which are classified to some known class and the number of known coins classified to a wrong known class. The assessment score is calculated by score = RC*25 + TP+TN - 100*(FP).

5 Experiments

We conducted some experiments to get good settings for the parameters. We found that for the angular/radial resolution 256×64 is a good trade-off betwees accuracy and speed of the classification. In general one can say that the higher the resolution the better the results, while the ratio $256/64 = 4$ seems to be optimal and should be kept fix. As already mentioned we use 6 discritization steps for the z-resolution.

In Table 5 we give some exemplary results. In the first column we conducted no additional reject criteria and considered no special cases. Further we show results for all three tranches with the additional reject criteria and special case treatment. Considering Tranche 1A we have, in fact, less false positive and only marginal shrinkage for TP+TN and hence also a higher assessment score. On all tranches, together 30000 coins, we have in total 7 coins which are wrongly classified to be known, while they are labeled as unknown. Six of these coins are very similar to known coin classes or are wrongly labeled. It does not seem possible to correct these mistakes. In Figure 7 we show three examplary errors. The only severe mistake is shown in Figure 7.a) and 7.b). In fact the coins do not show a high visual similarity and also the computed similarity values k are relatively small. But the nearest neighbors are searched in a list of about one hundred coins, which is a relativly small number. So it seems that the votes for both sides coincide accidently and hence the most powerful rejection criteria fails. Unfortunately the physical properties such as thickness and radius are also very similar, such that the confidence value is too big to let the final rejection work properly.

6 Conclusions

We presented a coin recognition system which is based on gradient directions only. The results show that the directional information is enough to build a reliable classification system while

a) b) c) d)

Figure 7: Front sides of wrongly classified coins. The 'Spielgeld'-coin shown in a) was confused with the coin shown in b). Images c) and d) show coins, which were labeled as unknown in the test set. But our system classified them to the coinclasses 3003 and 4002 respectivly, which are nearly identical to the coins shown in image c) and d).

the system is very robust to illumination and contrast changes. We have shown that the demand of a very low false positive rate is possible to reach. There might be several improvements of the system. Further fitting of the parameters like resolution, oversampling multiplier or smoothing width may improve the accuracy of the system. More sophisticated reject criteria and confidence values could also help to avoid false positives. Unfortunately the test set sizes of ten thousand coins are still to small to validate false positive rates of 0.01 percent.

References

[1] M. Nölle, H. Penz, M. Rubik, K. Mayer, I. Holländer and R. Granec. Dagobert - A new Coin Recognition and Sorting System. *In Proceedings of the 7th Internation Conference on Digital Image Computing - Techniques and Applications (DICTA'03), Syndney, Australia*

[2] D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13-2, 1981.

[3] J. Schulz, T. Schmidt, O. Ronneberger and H. Burkhardt. Fast Scalar and Vectorial Grayscale based Invariant Features for 3D Cell Nuclei Localization and Classification. *In Proceedings of the DAGM Symposium 2006, Berlin*

[4] S. Siggelkow and H. Burkhardt. Improvement of Histogram-Based Image Retrieval and Classification. *Proceedings of the International Conference on Pattern Recognition 3, 367-370, 2002*

[5] Haber, Ramoser, Mayer, Penz and Rubik. Classification of coins using an eigenspace approach. *In Pattern Recognition Letters (2005), Vol.26, No.1, 61-75*

[6] M. Fukumi, S. Omatu, F. Takeda and T. Kosaka. Rotation-invariant neural pattern recognition system with application to coin recognition. *IEEE Transactions in Neural Networks 3 (1992), No.2, 272-279*

[7] M. Adameck, M. Hossfeld and M. Eich. Three color selective stereo gradient method for fast topographic recognition of metallic surfaces. *In Proceedings of Electronic Imaging, Science and Technology, Machine Vision Application in Industrial Inspection XI, vol SPIE 5011, 2003, 128-139*