Parameter-learning for color sorting of bulk materials using genetic algorithms

Matthias Richter[1] and Jürgen Beyerer[1,2]

[1] Karlsruhe Institute of Technology, Institute for Anthropomatics and Robotics, Adenauerring 4, D-76131 Karlsruhe
[2] Fraunhofer Insitute of Optronics, System Technologies and Image Exploitation (IOSB), Fraunhoferstr. 1, D-76131 Karlsruhe

Abstract Sensor based sorting finds broad applications in mining, recycling and quality control. Digital image processing and pattern recognition are key components, as they determine whether to keep or discard an object under inspection. In many scenarios, the color of a material stands out as the primary sorting criterion. In this paper, we present a flexible system for color sorting of bulk materials based on semantic color features. The features are constructed in a three stages: the color occurrence frequencies of different materials are estimated and then fused to a small number of color classes, which in turn are used to map each color to a discrete attribute. A compact object descriptor composed of the fractions of foreground pixels that share the same attribute characterizes the objects under inspection. This descriptor has many advantages: it has a very clear, intuitive interpretation, is invariant to rotation and scale of the object and requires very little computation. However, a major drawback are the many variables that govern the construction process. Manual fine tuning requires a large amount of time and experience. Subtle changes in the parameters can have strong effects on the classification performance. To overcome this shortcoming, we propose a method to automatically learn the parameters by a genetic algorithm. We apply our method to wine grape sorting problems to show that this approach performs at least as good a human expert. At the same time, it takes considerably less effort on the human part and frees the operator to attend to other tasks.

1 Introduction

The applications of optical sensor based sorting range from mining of precious metals and minerals over recycling of synthetics to quality control of food stuffs. Digital image processing and subsequent pattern recognition are key components, as in this stage it is determined whether to keep or discard the objects under inspection [1]. The research community has developed a multitude of approaches leveraging methods from computer vision. Without going into too much detail[3], these approaches usually involve extraction of low to mid-level features such as hue histograms, Gabor-descriptors and shape models, which are then fed into machine learning algorithms to derive a sorting decision.

However, such methods are rarely found in commercially available systems. There are two main reasons for this: Firstly, their black box nature prevents the operators to change classification parameters when the sorting requirements change. Secondly, both the feature extraction and classification algorithm often require extensive computation, which is infeasible considering the run time requirements of automated visual inspection.

Instead, commercial systems often combine several simple rules that put thresholds on simple features. Scott, for example, sorts plastic waste into two fractions by measuring the ratio of absorbances at two wavelengths in the infrared spectrum and comparing it to a threshold [4]. Similarly, Lee and Anbalagan put multiple thresholds on the red, green and blue color channels, which result in multiple accept and reject zones in the color space [5]. An object is kept if the color of its foreground-pixels fall mostly into accept zones, otherwise it is discarded. More recently, Blasco et al. presented their system to sort pomegranate arils [6]. Instead of directly defining decision regions in the RGB color space, they use a single threshold on the average ratio between the red and green color channel. This simple approach performed comparably to LDA using the whole RGB tuple as feature vector. The advantage of these approaches is that they can be implemented in hardware, which enables very high sorting speeds. Furthermore, the system's sorting criteria can easily be adjusted by changing the thresholds. On the other

[3] A full review is out of the scope of this paper. Interested readers are instead referred to the encompassing surveys by Du and Sun [2] as well as Malamas et al. [3].

hand, initial investigation to find suitable features and thresholds is a laborious process that has to be carried out by a trained expert.

Hybrid approaches apply thresholds to high level features that are learned from the color distribution of the materials under inspection. This combines the easy set-up of machine-learning approaches with the flexibility of commercial solutions. Duffy et al. detect burn marks on air filters by estimating the probability whether a pixel shows a defect [7]. They derive a histogram that characterizes the color of burn marks by building the difference of RGB histograms of intact and defective samples. Defects are located by back-projection and thresholding the resulting image. In a follow-up publication, Bergasa et al. do not estimate histograms directly but instead model the color distribution of defects as a mixture of Gaussians [8]. While slower in training, this approach proves more robust in testing, as it accounts for underrepresented and unseen defect appearances. Explicit modeling the color-distribution is not always needed. For inspection of color tablets in pharmaceutical blisters, Derganc et al. find optimal decision boundaries in the color space by employing a mode seeking algorithm [9]. In training, an operator marks a pixel belonging to a tablet. Then, a labeling function is constructed by determining the corresponding mode and subsequent cluster growing.

Applications are not limited to classification though. Lee et al. grade the maturity of dates according to the color of the fruit's surface [10]. In their analysis, they found that the color of dates of different ripeness fall into a thin connected region in the RGB space. Consequently, they find a projection onto a one-dimensional manifold by solving a second order trivariate polynomial regression problem. In a later publication, Zhang, Lee et al. simplified the process by estimating a back-projection table mapping RG-values to a ripeness level [11]. The table is built by first collecting characteristic RG-histograms of four maturity grades and fusing the histograms into a single lookup-table. Missing entries are filled in by linear interpolation using neighboring values.

While all these approaches show good results with their respective product, application to different problem domains is questionable. The methods presented by Duffy et al. [7,8] leave the question how to handle multiple defect classes. The mode finding approach by Derganc et al. [9] works well when the surface of objects under inspection is relatively uniform in color, but may struggle when the objects' color distribution is multimodal. Lee's method [10] requires the color distribution of each class to be (approximately) supported on a one-dimensional connected manifold, which is seldom the case. The back-projection approach by Zhang et al. [11] alleviate this issue, but the construction of the lookup table is strictly tailored to grading date maturity and not directly applicable to other tasks.

With this in mind, we present a system capable of handling a large variety of products in different settings. The method is similar to the back-projection approaches presented above, but more general and not tailored to a specific product[4]. In a two-step process, we merge color histograms of different material fractions that may occur in the sorting application to color classes, which are then fused to build the lookup-table that maps RGB tuples to a discrete attribute. Objects are classified based on the fractions of pixels that map to each attribute.

2 Methods

In this section, we first describe our classification system and then turn our attention automatically learning the parameters that govern its behavior.

2.1 Classification System

Figure 10.1 shows the classification pipeline of our system. An input image $\mathcal{I}$ is transformed into an attribute image using a back-projection table that maps each RGB tuple $\mathbf{c} = (r, g, b)$ to an attribute,

$$\mathcal{A}(\mathbf{c}) = a \in \{-1, 0, 1, \ldots, M\}\,. \tag{10.1}$$

Here we use the convention that the attribute $\mathcal{A}(\mathbf{c}) = 0$ signifies a background pixel, whereas $\mathcal{A}(\mathbf{c}) = -1$ denotes an unknown color. Using blob analysis on the attribute image, single objects are extracted. A feature vector $\mathbf{m} = (m_{-1}, m_1, \ldots, m_M)^\top$ is calculated for each object, where each entry m_i in $\mathbf{m}$ represents the fraction of pixels that map to the i-th attribute. Formally, with $\mathcal{O}$ denoting the set of foreground pixels:

$$m_i = \frac{1}{|\mathcal{O}|} \sum_{(x,y)\in\mathcal{O}} \mathbb{1}[\mathcal{A}(\mathcal{I}(x,y)) = i]\,. \tag{10.2}$$

[4] In fact, we applied our method in different scenarios from recycling to food-inspection.

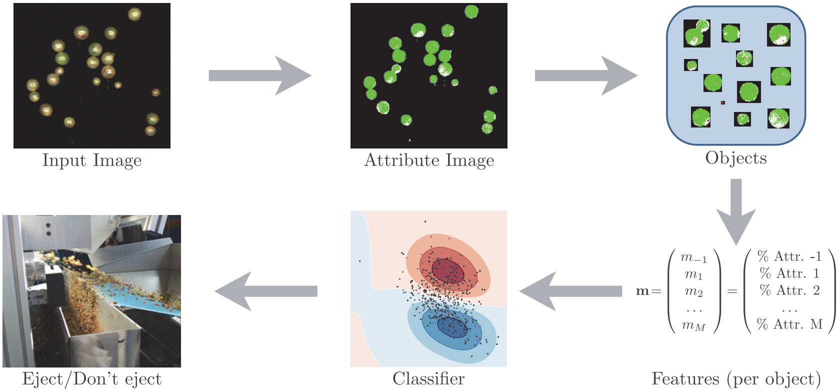


Figure 10.1: Overview of the classification pipeline.

The decision whether to keep or discard an object is done by a binary classifier $H(\mathbf{m}) = y \in \{-1, 1\}$ and a signal is sent to the actuating hardware. The choice of classifier is arbitrary, however keeping the intended application in mind, we typically settle for simple rule-based classifiers.

Color Attributes

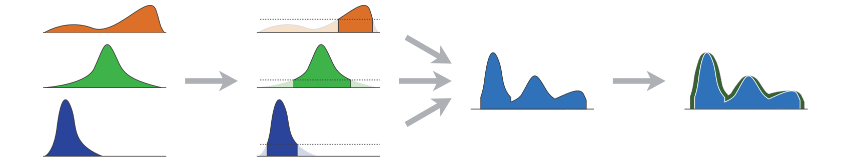
Key to this classification pipeline is the mapping from color to attribute in eq. (10.1). Figure 10.2 outlines the steps performed to derive color classes, which are the basic building blocks of the attribute-mapping $\mathcal{A}(\mathbf{c})$. In detail, the process is as follows:

The materials expected to be encountered during sorting are placed into the sorting machine and images are captured as if the system was in operation. From these images, color frequency estimates

$$\hat{p}_\kappa(\mathbf{c}|k), \quad k = 0, 1, \ldots, K \tag{10.3}$$

are collected. As with the attribute-mapping in eq. (10.1), the index $k = 0$ denotes the background. The remaining $\hat{p}_\kappa(\mathbf{c}|k)$ are estimated only from foreground pixels. The choice of estimator is arbitrary, but for the sake of simplicity we chose to use the joint RGB histogram.

If the ground-truth images show dirt particles or other non-target materials, the histograms may contain non-informative entries, which



(a) Frequency Estimates (b) Frequency Cutoff (c) Fused Color Class (d) Dilation Operation

Figure 10.2: Outline of the steps performed to derive color classes that are fused into the attribute table. Note that the figure shows 1D-histograms, while our system is based on dense, 3D-RGB histograms.

can be disruptive in later stages of the pipeline. Addressing this issue, we drop frequencies below a user-definable threshold $\beta \in [0, 1]$ (see Fig. 10.2 (b)). Formally, this amounts to the following operation:

$$\operatorname{cut}(\hat{p}_\kappa(\mathbf{c}|k), \beta) = \begin{cases} \frac{1}{Z}\,\hat{p}_\kappa(\mathbf{c}|k) & \text{if } \hat{p}_\kappa(\mathbf{c}|k) \geq \beta \\ 0 & \text{otherwise.} \end{cases} \tag{10.4}$$

Here Z is a normalization constant so that $\operatorname{cut}(\hat{p}_\kappa(\mathbf{c}|k), \beta)$ is a probability distribution. The resulting modified frequency estimates are fused into color classes (see Fig. 10.2 (c)) by weighted averaging,

$$\hat{p}_\mu(\mathbf{c}|m) = \sum_{k=0}^{K} \alpha_{km} \operatorname{cut}(\hat{p}_\kappa(\mathbf{c}|k), \beta_k), \quad m = 0, 1, \ldots, M, \tag{10.5}$$

where $\alpha_{km} \geq 0$ and $\sum_k \alpha_{km} = 1$. The color classes $\hat{p}_\mu(\mathbf{c}|m)$ represent higher-level features and correspond to semantic groups of materials.

The color frequency estimates and therefore the color classes are generally derived using a relatively small sample. As a consequence, the estimates may not accurately reflect the underlying distribution, especially when considering natural materials, whose appearance can fluctuate in time. Furthermore, external influences such as stray light can additionally alter the perceived color of the objects. To alleviate these problems, we apply a 3D morphological filter on the color classes:

$$\operatorname{dilate}(\hat{p}_\mu(\mathbf{c}|m), \delta) = \max_{\mathbf{o} \in [-\delta, \delta]^3} \hat{p}_\mu(\mathbf{c} - \mathbf{o}|m). \tag{10.6}$$

Finally, the attribute table is constructed by assigning the color class with the highest weighted probability for the given color,

$$\mathcal{A}(\mathbf{c}) = \begin{cases} -1 & \text{if} \max_m\{D_m\} = 0 \\ \arg\ \max_m \{\gamma_m\, D_m\} & \text{else,} \end{cases} \tag{10.7}$$

where $D_m = \mathrm{dilate}(\hat{p}_\mu(\mathbf{c}|m), \delta_m)$ and $\gamma_m \geq 0$. Incidentally, equation (10.7) can be interpreted as a maximum a posteriori classifier that assigns pixels to a color class, where γ_m encodes the class prior.

2.2 Parameter Learning

As mentioned in the previous section, the attribute table has a significant impact on the overall classification performance, more so than the classifier itself. However, finding the optimal configuration is a time consuming process where seemingly small adjustments can cause notably different sorting results. We therefore propose to automatically estimate good parameter combinations. Ideally, the user should only provide the initial color frequency estimates and the desired number of color classes M and the computer should figure out the remaining parameters.

One way to achieve this goal is to pose the task as optimization problem, i.e. to search the set of parameters that achieves the optimal classification performance.

Genetic Algorithms

Genetic algorithms (GA) are a well-known meta-heuristic to find a set of parameters $\boldsymbol{\theta}$ that maximize a fitness (or merit) function $f(\boldsymbol{\theta})$. GAs have been shows to work well with large or even infinite search spaces and are able to find the global optimum in non-convex problems [12]. However, the solution is generally only approximately optimal: With $\boldsymbol{\theta}^\star$ denoting the true optimum and $\varepsilon > 0$, a GA finds a solution $\hat{\boldsymbol{\theta}}$ with

$$|f(\boldsymbol{\theta}^\star) - f(\hat{\boldsymbol{\theta}})| < \varepsilon. \tag{10.8}$$

The method is modeled after the theory of natural selection: A population of individuals (possible solutions) produce offspring (new solution candidates) through recombination. The offspring is subject to random mutation and the fittest individuals are selected according to the fitness function f. The process repeats until a certain number of iterations is reached. The following pseudo-code outlines the approach:

```
**Input:** Population size $N$, Number of iterations $K$
**Output:** Candidate solutions $\mathcal{P}$

Randomly sample population $\mathcal{P} = \{\boldsymbol{\theta}_n | n = 1, \ldots, N\}$
**for** $k = 1$ **to** $K$:
    $\mathcal{C} \leftarrow \mathrm{crossover}(\mathcal{P})$
    $\mathcal{C} \leftarrow \mathrm{mutate}(\mathcal{C})$
    $\mathcal{P} \leftarrow \text{select-fittest}(\mathcal{P} \cup \mathcal{C})$
**return** $\mathcal{P}$
```

Key components are crossover and mutate operations, which explore the parameter-space around the existing solutions. There exists several alternatives to perform crossover, but here we focus on tournament selection with random recombination. In random recombination, the offspring $\boldsymbol{\theta}_c$ of two parent individuals $\boldsymbol{\theta}_p$ and $\boldsymbol{\theta}_q$ is produced by randomly choosing $\theta_c^{(i)}$ as the i-th element of either parent with equal probability. Each parent is selected by randomly sampling two candidates from the population and keeping the fitter one, i.e. for $\boldsymbol{\theta}_p$:

$$\boldsymbol{\theta}_p = \arg\,\max\{f(\boldsymbol{\theta}_{p_1}), f(\boldsymbol{\theta}_{p_2})\}, \quad \boldsymbol{\theta}_{p_1}, \boldsymbol{\theta}_{p_2} \in \mathcal{P}. \tag{10.9}$$

Since crossover alone is not sufficient to fully explore the parameter space, mutate performs a randomized local search by randomly changing elements of each $\boldsymbol{\theta}_c \in \mathcal{C}$. Finally select-fittest ranks all $\boldsymbol{\theta} \in \mathcal{P} \cup \mathcal{C}$ according to their fitness $f(\boldsymbol{\theta})$ and keeps only the N best-performing individuals.

The repeated application of crossover, mutation and selection have the effect that the population, which is initially scattered around the parameter space, converges onto a maximum of the fitness function. Due to the inherent randomization, GAs can recover from falling into local minima, which sets them apart from other methods such as gradient descent and hill-climbing. Furthermore, constraints on the parameters are almost trivial to implement by adjusting the crossover and mutate operations and regularization is achieved by adding an appropriate term to the fitness function.

Application

Due to the non-convex nature of our parameter-optimization and the large search space, we chose GAs to find good combinations of parameters. We constrain the parameters according to Section 2.1, i.e.

$$\alpha_{km}, \gamma_m \geq 0, \text{ with } \textstyle\sum_k \alpha_{km} = 1, \tag{10.10}$$

$$0 \leq \beta_k \leq 1 \text{ and} \tag{10.11}$$

$$\delta_m \in \mathbb{N}_0. \tag{10.12}$$

As fitness function we employ Matthews correlation coefficient [13], which can be interpreted as the correlation coefficient between ground-truth and prediction of the classifier $H(\mathbf{m})$. With n_{tp}, n_{fp}, n_{tn} and n_{fn} denoting the number of true positive, false positive, true negative and false negative classifications on a validation sample it can be defined as

$$\text{MCC} = \frac{n_{\text{tp}}\, n_{\text{tn}} - n_{\text{fp}}\, n_{\text{fn}}}{\sqrt{(n_{\text{tp}} + n_{\text{fp}})(n_{\text{tp}} + n_{\text{fn}})(n_{\text{tn}} + n_{\text{fp}})(n_{\text{tn}} + n_{\text{fn}})}}. \tag{10.13}$$

3 Experiments

We validated our approach by comparing the classification performance of hand tuned parameters to parameters learnt by the GA. We considered the sorting problem of discriminating healthy wine berries from grapes with fungal infection and unwanted parts of the plant. Experiments were performed with three varieties of wine: Pinot noir, Pinot blanc and Riesling. All images were acquired using an off-the-shelve RGB line-scan camera. Since berries of the Pinot noir variety are very dark and show low contrast to the black background, the blue channel was replaced with a NIR-channel in this case. To reduce the parameter space, we do not perform frequency thresholding (i.e. set $\beta_k = 0$ for all k) and constrain the background class to not include, and not to be included in the other color classes by setting $\alpha_{00} = 1$ and $\alpha_{0m} = \alpha_{k0} = 0$. Parameters were mutated by each selecting a random α_{km}, γ_m and δ_m and assigning a new value in the variable's domain with probability of $p = 0.8$, $p = 0.5$ and $p = 0.3$ respectively. We chose a linear SVM as classifier $H(\mathbf{m})$, since it is relatively fast to train and evaluate, and therefore reduces the time required for the optimization.

Grape variety	# of samples		MCC wrt. selection method	
	positive	negative	manual	learned
Pinot noir	2416	641	0.47	0.70 ±0.12
Pinot blanc	332	291	0.86	0.86 ±0.17
Riesling	1061	235	0.88	0.86 ±0.11

Table 10.1: Overview over the results of our experiments.

Table 10.1 shows the results of our experiments. In case of learned parameters, we performed stratified 10-fold cross-validation to estimate the mean and standard deviation of MCC values. We did not perform cross-validation with hand-tuned parameters. In each fold, the training set was randomly split in two subsets, where the first one was used to estimate parameters and the second was used to train the classifier. Especially with the Pinot blanc variety, this resulted in very few training samples and subsequently relatively high variance of the classification performance. Nonetheless, the classification performance is on par with manually estimated parameters. In case of Pinot noir, the GA approach even outperformed the human expert, who had difficulties finding an appropriate set of parameters due to the (apparent) lack of an informative blue channel. Table 10.2 shows the learned parameters of the best performing solution with the Pinot noir variety. While the first color class puts more emphasis on the third and fourth color frequency estimate (defects), the second color class is influenced primarily by the fifth and sixth frequency estimate (ripe berries). This roughly corresponds to $m = 1$ and $m = 2$ denoting *defect*- and *accept*-classes. However, both classes consider all available color estimates (apart from the background – which was enforced by the experimental constraints). This contrasts with the approach of a human expert, who typically selects few color frequency estimates to build the color classes.

4 Conclusion

In this paper we have presented both a method for color classification of bulk materials and a method to automatically estimate the parameters governing the classification. The system is flexible and can accommo-

α_{km}	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	γ_m	δ_m
$m=0$	1.00	0.00	0.00	0.00	0.00	0.00	0.06	1
$m=1$	0.00	0.16	0.27	0.47	0.09	0.01	1.00	3
$m=2$	0.00	0.07	0.01	0.18	0.41	0.34	0.08	3

Table 10.2: Best performing learned configuration for the Pinot blanc variety.

date a wide range of materials. While manual set-up is a labor intensive and time-consuming process, parameter learning only requires human interaction when estimating the basic color distributions. The remaining learning process is fully automatic and frees the human to attend to other tasks. Moreover, it requires no knowledge of the inner workings of the systems and therefore allows non-experts to bootstrap a working classifier.

However, there is still room for improvement. In particular, one could use kernel density estimators instead of joint RGB histograms to derive the color frequencies $\hat{p}_\kappa(\mathbf{c}|k)$. Doing so would remove the need to perform dilation (10.6) and therefore reduce the dimensionality of the parameter space. Another modification would be to encourage sparsity of the α_{km} by including an appropriate term in the fitness function. The resulting parameters would be more similar to configuration set up by a human expert and thus be more open to interpretation. Lastly, the genetic algorithm could be replaced by other heuristic optimization procedures such as particle swam optimization, which puts a stronger emphasis on local search, or simulated annealing, which evaluates the merit function less frequently than a GA and thus might be faster to find the optimum.

References

1. H. Wotruba and H. Harbeck, “Sensor-based sorting,” *Ullmann's Encyclopedia of Industrial Chemistry*, 2010.
2. C.-J. Du and D.-W. Sun, “Learning techniques used in computer vision for food quality evaluation: a review,” *Journal of Food Engineering*, vol. 72, no. 1, pp. 39–55, Jan. 2006.

3. E. N. Malamas, E. G. Petrakis, M. Zervakis, L. Petit, and J.-D. Legat, "A survey on industrial vision systems, applications and tools," *Image and Vision Computing*, vol. 21, no. 2, pp. 171–188, Feb. 2003.
4. D. M. Scott, "A two-colour near-infrared sensor for sorting recycled plastic waste," *Measurement Science and Technology*, vol. 6, no. 2, pp. 156–159, Feb. 1995.
5. D.-J. Lee and R. S. Anbalagan, "High-speed automated color-sorting vision system," in *Optical Engineering Midwest*, vol. 2622. International Society for Optics and Photonics, Aug. 1995, pp. 573–579.
6. J. Blasco, S. Cubero, J. Gómez-Sanchís, P. Mira, and E. Moltó, "Development of a machine for the automatic sorting of pomegranate (Punica granatum) arils based on computer vision," *Journal of Food Engineering*, vol. 90, no. 1, pp. 27–34, Jan. 2009.
7. N. Duffy, J. Crowley, and G. Lacey, "Object detection using colour," in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 1. IEEE Comput. Soc, 2000, pp. 700–703.
8. L. Bergasa, N. Duffy, G. Lacey, and M. Mazo, "Industrial inspection using Gaussian functions in a colour space," *Image and Vision Computing*, vol. 18, no. 12, pp. 951–957, Sep. 2000.
9. J. Derganc, B. Likar, R. Bernard, D. Tomaževič, and F. Pernuš, "Real-time automated visual inspection of color tablets in pharmaceutical blisters," *Real-Time Imaging*, vol. 9, no. 2, pp. 113–124, Apr. 2003.
10. D.-J. Lee, J. K. Archibald, Y.-C. Chang, and C. R. Greco, "Robust color space conversion and color distribution analysis techniques for date maturity evaluation," *Journal of Food Engineering*, vol. 88, no. 3, pp. 364–372, 2008.
11. D. Zhang, D.-J. Lee, B. J. Tippetts, and K. D. Lillywhite, "Date maturity and quality evaluation using color distribution analysis and back projection," *Journal of Food Engineering*, vol. 131, pp. 161–169, Jun. 2014.
12. M. Mitchell, *An introduction to genetic algorithms*, 1998.
13. B. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochimica et Biophysica Acta (BBA) - Protein Structure*, vol. 405, no. 2, pp. 442–451, Oct. 1975.