Automated Classification of Bitmap Images Using Decision Trees

Pavel Surynek[1], Ivana Lukšová[1]

[1]Charles University in Prague, Faculty of Mathematics and Physics
Department of Theoretical Computer Science and Mathematical Logic
Malostranské náměstí 25, Praha, 118 00, Czech Republic
pavel.surynek@mff.cuni.cz, ivana.luksova@gmail.com

Abstract. The paper addresses the design of a method for automated classification of bitmap images into classes described by the user in natural language. Examples of such naturally defined classes are images depicting buildings, landscape, artistic images, etc. The proposed classification method is based on the extraction of suitable attributes from a bitmap image such as contrast, histogram, the occurrence of straight lines, etc. Extracted attributes are subsequently processed by a decision tree which has been trained in advance. A performed experimental evaluation with 5 classification classes showed that the proposed method has the accuracy of 75%-85%. The design of the method is general enough to allow the extension of the set of classification classes as well as the number of extracted attributes to increase the accuracy of classification.

Keywords: image classification, attribute extraction, decision trees, learning

1 Introduction and Motivation

As digital cameras (industrial or personal) are still more common, the demand for tools for the automated processing of recorded data increases. The large amount of data precludes its manual processing and represents the main reason for the requirement on automation. The typical situation many users of personal digital cameras often face is assorting several hundreds or even thousands of pictures from holiday which typically becomes a tedious and time consuming task. Similar tasks arise in industrial applications where it is often necessary to categorize automatically recorded images – for instance images from a surveillance camera that contain an interesting activity need to be distinguished from uninteresting ones.

This work is devoted to an analysis of static images - that is, *bitmap images*. The basic assumption is that the source of bitmap images is not restricted in any way. Thus they can be represented by photographs recorded by the digital camera or by artificially created images such as rendered images. The particular task we are solving in this work is the *classification of bitmap images* into a set of classes defined by an expression in natural language. These classes are represented for example by images depicting landscapes or buildings recorded by the digital camera, or by the artistic images created by an artificial process.

The primary goal was to design a completely automated method that decides what class an input bitmap image belongs to. The set of classes for classification is known in advance (that is, they are not a part of the input). One bitmap image can belong into multiple classification classes (not just one). The secondary goal was to develop such a method which is general enough to allow extending the set of classes for classification as well increasing the accuracy of the process of classification. It is expectable that the prototype implementation will have low accuracy, thus it is necessary to allow further improvements of the method directly by the initial design.

The concept of *decision tree* [6, 8, 11] has been chosen as the basic building block of the method. This choice was guided by the initial analysis of the problem. The decision tree is a structure that describes assignment of a certain value to an input vector of attributes. The assigned value is represented by a classification class in our case and the input vector of attributes consists of properties extracted from the bitmap image such as *brightness*, *histogram* [1] or the *occurrence of straight lines*. It is supposed that all the attributes used for classification are expressed using the integer value. The sufficiently accurate classification using the decision tree is implied by determining a suitable set of attributes extracted from the image in this context. It is necessary to find attributed that characterize the image well with respect to the set of classification classes. It is evident that the choice of attributes has the great impact on the method. On the other side, it provides a great potential to increase the accuracy of the method.

The secondary goal of our design is satisfied by the concept of decision tree as well. The decision tree works completely automatically in the deciding mode (that is, no user interaction is required). The extension of the decision tree with more classification classes and attributes is a routine augmentation in fact.

This article is based on results elaborated within [4]. The text is organized in the following way. The classification task is formally introduced first. Then initial classification classes we need to be classified by the method are described. Techniques from artificial intelligence and computer graphics that are exploited by our classification method and the classification method itself are described in the next section. The last section is devoted to an experimental evaluation of the prototype implementation of the method written in the Java language on a large set of testing images from different sources.

2 The Task of Bitmap Classification

Let K be a finite set of classification classes and let $\mathcal{I}$ be a set of bitmap images. Each classification class $t \in \mathcal{K}$ has assigned a description in the natural language $d(t)$. Next, let a function $c: \mathcal{I} \longrightarrow 2^{\mathcal{K}}$ defines an assignment of classification classes to bitmap images so that $\forall I \in \mathcal{I}\ \forall t \in c(I)\ d(t)$ describes the image I well from the user's perspective. Notice that the function c is not known in explicit form. It is known implicitly by the user when she or he can give $c(I)$ for an individual input image. There may be also multiple users who together agree on c by some negotiation strategy.

The following definitions provide us with the precise formulation of the classification task and the demands placed on the classification method.

Definition 1 (*classification task, classification method*). A *classification method* is an algorithm that computes a function $\boldsymbol{c'}: \boldsymbol{\mathcal{I}} \longrightarrow \boldsymbol{2^{\mathcal{K}}}$. A *classification task* with bitmap images corresponds to calculation of $\boldsymbol{c'(I)}$ for a given input bitmap image $\boldsymbol{I} \in \boldsymbol{\mathcal{I}}$. □

The classification method as defined above can be completely mistaken with respect to the user's opinion. Therefore we have the following definition allow us to distinguish the accuracy of classification.

Definition 2 (*accuracy of the classification method*). The classification method corresponding to the function $c': \mathcal{I} \longrightarrow 2^{\mathcal{K}}$ is *accurate* for a classification class $t \in \mathcal{K}$ and an input bitmap image $I \in \mathcal{I}$ if and only if $t \in c'(I) \Leftrightarrow t \in c(I)$. Let $\mathcal{S} \subset \mathcal{I}$ be a finite set of bitmap images and let $\mathcal{S}' \subseteq \mathcal{S}$ be a maximum subset of images, where the method is accurate for the classification class $t \in \mathcal{K}$ (that is, there is no such a set $\mathcal{S}'' \supset \mathcal{S}'$ for which the method is accurate with respect to $t \in \mathcal{K}$), then $|\mathcal{S}'|/|\mathcal{S}|$ is called an *accuracy* of the classification method for the classification class t with respect to the set of images $\mathcal{S}$. □

The above definition of the classification task captures large flexibility in terms of extensibility and scalability. Observe that for instance description of classification classes can be represented by a keyword or a set of keywords. Moreover, the mentioned user's point of view can be determined by a preference of users within a social network (such as *flickr* [15] or similar) who can annotate images with some predefined descriptions in the natural language.

The goal of our work is of course to design a classification method with accuracy as high as possible with respect to the given set of classification classes and for the greatest possible set of input bitmap images (ideally, for all the images that the user can submit in the input).

2.1 Initially Selected Classification Classes

As the initial goal we required the method to manage the classification task with respect to the following set of five classification classes $\mathcal{K} = \{P, A, B, L, M\}$. These classes were inspired by requirements of many owners of digital cameras who want to automatically classify their databases of images. Let us emphasize that the set of classification classes $\mathcal{K}$ is used to prove the concept. A larger set of classes is supposed to be used in a real life application of the method.

In the following text, each class $t \in \mathcal{K}$ is described using a brief natural language expression $d(t)$. Images that belong to the class and those that do not are mentioned. Examples of positively and negatively classified images with respect to proposed classes are also shown.

P: photography

$d(P)$ ="a photography created by a digital or analog camera"

Positive and negative examples of bitmap images with respect to the class P are shown in Fig. 1. Artistic images that are artificially created do not belong to this clas-

sification class for instance. Images created by an analog camera scanned into digital format are also supposed to belong into this class.

Fig. 1. A bitmap image classified into the class ***photography*** **(P)** [left part] and an example of an image not classified into this class [right part] (an *artistic image* (A)).

A: artistic image

$d(I)$ ="an image created using a drawing or a painting technique"

An example of artistic bitmap image is shown in the right part of Fig. 1 and in Fig. 2. For instance, photography does not belong to this class. Pictures with diagrams/schemes and rendered images do not belong to this class as well. On the other side, a painting/drawing technique simulated in a computer program is acceptable for this class.

Fig. 2. An image classified into the class ***artistic image*** **(A)**.

B: buildings

$d(B)$ ="an image depicting a building or some kind of architecture"

An example of an image classified into the class B is shown in Fig. 3. A photograph depicting some kind of building is typical representative. To give a negative example, landscapes do not belong to this class. However, building located in landscape represents a positive example with respect to this class.

L: landscapes

$d(L)$ ="an image depicting a landscape"

One of examples of images depicting landscape is shown in Fig. 4. Images of artificial objects are not classified into this class. On the other side, an image of artificial object placed in the landscape can represent a landscape as the whole so it is classified positively.

M: macro objects
$d(B)$ ="a photography of an object from the small distance with blurred background"

An example of image with macro object is shown in Fig. 5. The center of the image contains the target object while the background is blurred. Images of flowers are the most common candidates for macro object. On the contrary, artistic images even though they depict flowers are not classified as macro objects.

Fig. 3. An example of the image classified by the user into the class ***buildings*** (**B**).

Notice that the textual description $d(t)$ of the classification class $t \in \mathcal{K}$ in the natural language deals with the positive specification of the given class (that is, how does the positive example look like). Negative specification of the class t is given by the situation, that the natural language description does not describe a given image well. Observe further, that the proposed set of classification classes allow for instance the existence of a bitmap image $I_1 \in \mathcal{I}$ such that $c(I_1) = \{A, B, L\}$. On the other side, there cannot exist an image $I_2 \in \mathcal{I}$ such that $c(I_2) = \{P, A\}$.

Fig. 4. Bitmap image classified into the class ***landscapes*** (**L**).

Fig. 5. An example of bitmap image classified as ***macro object*** (**M**).

3 The Initial Analysis of the Problem

During the development of the classification method, several key questions needed to be answered. First, it was not clear how to formally handle the condition that $\forall I \in \mathcal{I}$ $\forall t \in c(I)$ $d(t)$ describes the image I well from the user's perspective. It is necessary to take into account that $c: \mathcal{I} \longrightarrow 2^{\mathcal{K}}$ is defined by the particular user or the set of users and hence can be biased somehow. Thus we need to design the method so that is should be configurable with respect to different users. Next, it is necessary to take into account that function c is not known explicitly.

Finally, it turned out to be easiest to record function $c: \mathcal{I} \longrightarrow 2^{\mathcal{K}}$ partially using an annotated training set of selected images. This solution takes into account the individual users as well as the fact that knowledge of c is implicit only (that is, we can put queries on c but are unable to enumerate it as the whole). The classification method itself implementing assignment $c': \mathcal{I} \longrightarrow 2^{\mathcal{K}}$ should be identical with c on all (or on the almost all) input images from the training set. Additionally it should be as much accurate as possible on other images with respect to the function c. To satisfy these requirements the method should have a good **generalization** ability [12].

We decided to use the concept of *decision tree* [8, 11, 12] as the core of our classification method. There exist many successful learning algorithms for decision trees that support its generalization abilities. As it has been mentioned, the decision tree can be easily extended for a larger set of classification classes as well as attributes.

The next important question is the identification and acquiring attributes that characterizes the bitmap images with respect to classification classes well. These attributes are to be used by the decision tree. Intuitively, the following categories of attributes were suggested: *basic color characteristics* (number of colors, relative occurrence of colors, etc.), characteristics based on the *histogram* (contrast, local contrast), and characteristics based on *edge detection* [5, 13, 18] (occurrence of straight lines, occurrence of right angles).

The basic assumption was that basic color characteristics can help to distinguish photographs from artistic images. Next it was expected that characteristics based on histogram can help to distinguish macro object from landscapes that differ in contrast sharply. Finally, characteristics based on edge detection can distinguish images of artificial object from natural ones.

In the complete classification all the attributes play some role. Interestingly, the importance of individual attributes is determined by the learning process of the decision tree.

4 Techniques for the Process of Classification

One of the major contributions of this work is the suggestion of a set of attributes suitable for image classification and the design of methods for their extraction. The following text is devoted to the existent concept of decision tree – a brief description is given. The process of extraction of attributes is described in more details since it is the original contribution of this work.

4.1 The Concept of Binary Decision Tree

Suppose that we have a set of attributes $\mathcal{P}$ extracted from the bitmap image. Each attribute can be assigned a value from the given domain (integers or floating point numbers). Next suppose that we have a classification class $t \in \mathcal{K}$. A *binary decision tree* for the class t is an oriented tree where each node except leaves have two successors. The root and internal nodes are assigned an attribute $p \in \mathcal{P}$. Leaves are assigned

the Boolean value $False$ or $True$. Edges from a node which is assigned an attribute p are assigned disjoint subsets of the set of values for the attribute p.

The decision tree assigns the indication of the membership into the given classification class t to a given vector of attributes. Having a vector of attributes, the indication of the membership into the given class is determined by a *controlled traversal* of the decision tree from the root to the leaf. The controlled traversal starts in the root and is defined inductively. Suppose that the current node is assigned an attribute p. The traversal then continues to the endpoint of the edge that starts in the current node and is assigned a subset of values which the value of the attribute p from the input vector belongs to.

The decision tree is said to be *complete*, if the controlled traversal is defined for every input vector of attributes. To represent classification function $c: \mathcal{I} \longrightarrow 2^{\mathcal{K}}$ we will need $|\mathcal{K}|$ decision trees. Observe that the accuracy of the decision tree is now a well defined concept.

The process of leasing of the decision tree exploited in our method is based on the algorithm ID3 [8]. Although ID3 is outperformed by C4.5 [9] we used ID3 for scholarly purposes. An example of decision tree is shown in Fig. 6.

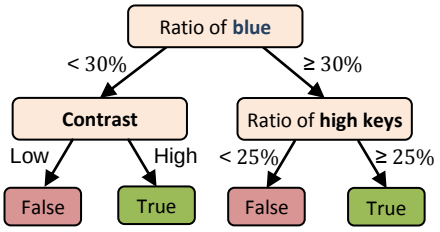

Fig. 6. An example of a simple decision tree for image classification with respect to the classification class ***landscapes*** (**L**).

4.2 The Extraction of Attributes from Colors

In the following text we will be working with bitmap images of 8-bit depth for each color component [1, 14]. The eventual adaptation for the different color depth is straightforward. The basic characteristic which will be extracted from the image is based on the color information. The following definition formalizes the concept of color information.

Definition 3 (*bitmap image – color, monochrome*). A *color bitmap image* is a 5-tuple $I_{rgb} = (r, g, b, x_{max}, y_{max})$ where $r: \{0,1,\dots,x_{max}\} \times \{0,1,\dots,y_{max}\} \longrightarrow \{0,1,\dots, 255\}$, $g: \{0,1,\dots,x_{max}\} \times \{0,1,\dots,y_{max}\} \longrightarrow \{0,1,\dots,255\}$, and $b: \{0,1,\dots, x_{max}\} \times \{0,1,\dots,y_{max}\} \longrightarrow \{0, 1,\dots,255\}$ are functions that assign individual pixels of the image the value of *red*, *green*, and *blue* component respectively; x_{max} represents the horizontal size of the image, y_{max} represents the vertical size of the image (the size of the image is thus $(x_{max}+1) \times (y_{max}+1)$ pixels). *A monochrome bitmap image* is a triple $I_{mono} = (i, x_{max}, y_{max})$, where $i: \{0,1,\dots,x_{max}\} \times \{0,1,\dots,y_{max}\} \longrightarrow \{0,1,\dots,255\}$ is a function that assigns the individual pixels of the image their intensity. For a color bitmap image the corresponding monochrome image can be obtained using the following expression: $(x,y) = 0.299\, r(x,y) + 0.587\, g(x,y) + 0.114\, b(x,y)\ \ \forall x \in \{0,1,\dots,x_{max}\};\ \forall y \in \{0,1,\dots,y_{max}\}$ (this expression has been

determined empirically for humans [1, 14]). The size of the image is defined as the number of its pixels, that is $s_{xy} = (x_{max} + 1) * (y_{max} + 1)$. □

A useful characteristic is the total *number of colors* of the image. Having an artistic image which is not affected by any noise it is expectable that the total number of colors will be low. The total number of colors is given by the expression: $c_{rgb} = |\{(r(x,y), g(x,y), b(x,y))|x \in \{0,1,\dots,x_{max}\} \wedge y \in \{0,1,\dots,y_{max}\}\}|$.

The next interesting characteristic is represented by the *occurrence of the individual colors*. As the total number of possible colors is too high it is necessary to restrict on a certain color palette [1, 14] and calculate the occurrence of colors that are close to the colors from the palette.

The easiest way how to create a color palette is to represent colors (r, g, b), where $\frac{\alpha}{\eta}256 \leq r < \frac{\alpha+1}{\eta}256 \wedge \frac{\beta}{\eta}256 \leq g < \frac{\beta+1}{\eta}256 \wedge \frac{\gamma}{\eta}256 \leq b < \frac{\gamma+1}{\eta}256$ for $\eta \in \mathbb{N}$ and $\alpha, \beta, \gamma \in \{0,1,\dots,\eta-1\}$ using a single color. Since we require the total number of colors in the palette to be low, the value of the parameter η should be low as well. In the implementation of the method we used $\eta = 4$. Occurrence of individual colors is the defined by a function: $rgb(\alpha,\beta,\gamma) = |\{(x,y)|x \in \{0,1,\dots,x_{max}\} \wedge y \in \{0,1,\dots,y_{max}\} \wedge \frac{\alpha}{\eta}256 \leq r(x,y) < \frac{\alpha+1}{\eta}256 \wedge \frac{\beta}{\eta}256 \leq g(x,y) < \frac{\beta+1}{\eta}256 \wedge \frac{\gamma}{\eta}256 \leq b(x,y) < \frac{\gamma+1}{\eta}256\}|/s_{xy}$, where $\eta \in \mathbb{N}$ and $\alpha, \beta, \gamma \in \{0,1,\dots,\eta-1\}$.

A *typical palette* for each classification class has been determined using the function rgb - it consists of the most frequent colors. The attribute used for classification is represented by the difference of the occurrence of colors in the input image from the typical palette of the individual classification classes.

4.3 Attributes Exploiting Contrast and Histogram

Another important characteristic for bitmap images is represented by the *histogram* [1, 14]. In photographic techniques there are frequently used terms as *low-key*, *mid-key*, and *high-key* photograph. These terms express the high relative occurrence of dark colors, middle-tone colors, and light colors respectively. We will quantify these terms exactly using the histogram.

Definition 4 (*histogram*). A *histogram* for the given monochrome image $I_{mono} = (i, x_{max}, y_{max})$ is a function $h_i: \{0,1,\dots,255\} \longrightarrow \mathbb{N}_0$ (natural numbers including zero), where $h_i(j) = |\{(x,y)|x \in \{0,1,\dots,x_{max}\} \wedge y \in \{0,1,\dots,y_{max}\} \wedge i(x,y) = j\}|$. For the color image $I_{rgb} = (r, g, b, x_{max}, y_{max})$ we define the histogram in an analogical way using a triple of functions $h_r: \{0,1,\dots,255\} \longrightarrow \mathbb{N}_0$, $h_g: \{0,1,\dots,255\} \longrightarrow \mathbb{N}_0$, and $h_b: \{0,1,\dots,255\} \longrightarrow \mathbb{N}_0$ for individual color components. □

An important characteristic connected with the histogram is represented by the *contrast*. We consider an image with the great difference between dark and light colors as the contrast one. The histogram of such an image is not concentrated in the narrow interval of the intensity.

Table 1. Several characteristics based on the histogram.

Characteristic	Calculation
Ratio of low colors $r_{low}^{h_i}$	$r_{low}^{h_i} = \frac{1}{s_{xy}} \sum_{j=0}^{85} h_i(j)$
Ratio of mid colors $r_{mid}^{h_i}$	$r_{mid}^{h_i} = \frac{1}{s_{xy}} \sum_{j=86}^{170} h_i(j)$
Ratio of high colors $r_{high}^{h_i}$	$r_{high}^{h_i} = \frac{1}{s_{xy}} \sum_{j=170}^{255} h_i(j)$
Contrast by **Michelson** $c_M^{h_i}$	$c_M^{h_i} = \frac{i_{max} - i_{min}}{i_{max} + i_{min}}$
Contrast by **Weber** $c_{rms}^{h_i}$	$c_{rms}^{h_i} = \sqrt{\frac{1}{s_{xy}} \sum_{x=0}^{x_{max}} \sum_{y=0}^{y_{max}} (i(x,y) - i_\mu)^2}$

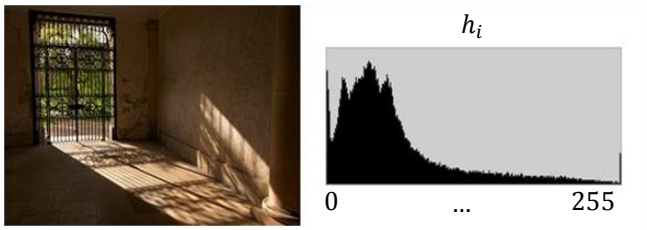

Fig. 7. An example of *low-key* image and its histogram.

The contrast can be formally defined in various ways. In the design of the classification method we used definition from [7], that is definitions by Michelson and Weber. To introduce these concepts we need to know the value of the minimum, maximum, and the average intensity which are defined by the following expressions respectively: $i_{min} = \min \{j | j \in \{0,1,\dots 255\} \wedge\ h_i(j) > 0\}$, $i_{max} = \max \{j | j \in \{0,1,\dots 255\} \wedge\ h_i(j) > 0\}$, and $i_\mu = \frac{1}{s_{xy}} \sum_{j=0}^{255} (j * h_i(j))$.

Several characteristics based on the histogram for the intensity are shown in Table 1. Notice that all the mentioned characteristics can be defined for individual color components as well (however, the experimental implementation does not use them).

Sometimes it is necessary to use a so called *local contrast*. The local contrast can be found on photographs of objects photographed from the small distance (*macro objects*). The high contrast is concentrated in the central part of the image typically (the object itself) while other parts of the image have the low contrast (background). In our classification method, the input image has been divided by the grid with squares of the size of 25×25 pixels while the contrast (by Michelson $c_M^{h_i}$ as well as by Weber $c_{rms}^{h_i}$) has been calculated and used as an attribute for every square of the grid.

4.4 Attributes Based on Edge Detection

The core technique which allows us to distinguish the class *buildings* (**B**) from other classes exploits *edge detection* [5, 18]. As the extraction and application of information about edges represents the most complex technique developed within our work, we will describe it more formally using algorithms in pseudo-code.

The goal is to obtain the explicit information about edges occurring in the input bitmap image. This explicit knowledge then allows calculating characteristics such as the *occurrence of right angles.*

Algorithm 1. *Algorithm of* ***edge detection***. The input is a monochromatic image $\boldsymbol{I_{mono}}$ and a convolution matrix $\boldsymbol{h}$. The output is a monochromatic image with highlighted detected edges.

function *Detect-Edges* $(I_{mono} = (i, x_{max}, y_{max}), h, k_{max})$: **grayscale image**
for $x = 0,1, \dots, x_{max}$ **do**
  for $y = 0,1, \dots, y_{max}$ **do**
    $s \leftarrow 0$
    for $k_x = -k_{max}, \dots, -1,0,1, \dots, k_{max}$ **do**
      for $k_y = -k_{max}, \dots, -1,0,1, \dots, k_{max}$ **do**
        $s \leftarrow s + h(k_{max} + k_x + 1, k_{max} + k_y + 1) * i(x + k_x, y + k_y)$
    $i_E(x, y) \leftarrow s$
return (i_E, x_{max}, y_{max})

The first step consists in using the standard method for **edge detection** based on convolution which the result is the set of edges in the implicit form (edges are merely highlighted in the image). Particularly, the convolution with the *Laplace operator* [1, 14] has been used where the internal convolution matrix is as follows: $h = ((1,1,1); (1,-8,1); (1,1,1))$.

The process of edge detection is formally described using pseudo-code as Algorithm 1. The next phase is the transformation of the implicit information about edges into the **explicit** one. This step is done by a so called *Hough transformation* [3, 10]. The transformation algorithm calculates all the lines that goes through every highlighted point (that can be recognized according to its intensity which is higher than the given threshold θ_H) and are expressed by the equation $\rho = x * \cos \vartheta + y * \sin \vartheta$. More precisely, the parameters ϑ and ρ are calculated for every highlighted point (x, y). As there is infinite number of solutions we use the discrete sampling of the parametric space.

The occurrence of the line is then indicated by the local maximum of the function depending on the parameters ϑ and ρ which expresses the number of highlighted points on the line corresponding to the parameters. The detected lines are subsequently segmented into the line segments according to the intensity of pixels on the line. If the intensity is not high enough the line is segmented (again, the intensity lower than the given threshold θ_S indicates line segmentation). The result is the set of line segments $L_S = \{((a_x^1, a_y^1), (b_x^1, b_y^1), (\varphi^1, \rho^1)), \dots, ((a_x^n, a_y^n), (b_x^n, b_y^n), (b_x^n, b_y^n), (\varphi^n, \rho^n))\}$, where (a_x^k, a_y^k) is the start point, (b_x^k, b_y^k) is an endpoint of the k-th line segment, and (φ^k, ρ^k) are parameters of the line on that the k-th line segment lies.

Using the explicit knowledge about line segments it is possible to define various characteristics. Some of them are shown in Table 1. We can then calculate the occurrence of right angles which is important for classification of buildings (B). The process of right angle detection is shown as Algorithm 3. The total number of detected right angles is used as the decision attribute, that is the value $c^{A_R} = |A_R|$.

Algorithm 2. *Modified* ***Hough transformation***. The input is a monochromatic image I^e_{mono}, obtained by edge detection. The output is a set of line segments in the explicit form.

function *Hough-Transformation* $(I^e_{mono} = (i^e, x^e_{max}, y^e_{max}), \varphi_{max}, \rho_{max}, \theta_H, \theta_S)$: **set**
$a_H \leftarrow \vec{0}$
for $x = 0,1, \dots, x^e_{max}$ **do**
  for $y = 0,1, \dots, y^e_{max}$ **do**
    if $i^e(x,y) \geq \theta_H$ **then**
      for $\varphi = 0,1, \dots, \varphi_{max}$ **do**
        $\rho \leftarrow x * \cos((\varphi/\varphi_{max})\pi) + y * \sin((\varphi/\varphi_{max})\pi)$
        $a_H(\varphi, \rho) \leftarrow a(\varphi, \rho) + 1$
$L \leftarrow \emptyset$
for $\varphi = 0,1, \dots, \varphi_{max}$ **do**
  for $\rho = 0,1, \dots, \rho_{max}$ **do**
    if a_H gains local mixmum in (φ, ρ) **then**
      $L \leftarrow L \cup \{(\varphi, \rho)\}$
$L_S \leftarrow \emptyset$
for each $(\varphi, \rho) \in L_S$ **do**
let $[(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)]$ is a sequence of (integer) points of line (φ, ρ) in the image I^e_{mono} sorted along the line
  $(a_x, a_y) \leftarrow \perp$
  for $k = 1,2, \dots, m$ **do**
    if $(a_x, a_y) = \perp$ **then**
      if $i_E(x_k, y_k) \geq \theta_S$ **then**
        $(a_x, a_y) \leftarrow (x_k, y_k)$
    else
      if $i_E(x_k, y_k) < \theta_S$ **then**
        $(a_x, a_y) \leftarrow \perp$
        $L_S \leftarrow L_S \cup \{((a_x, a_y), (x_k, y_k), (\varphi, \rho))\}$
return L_S

Algorithm 3. *Detection of* ***right angles***. The input is a set of line segments $\boldsymbol{L_S}$ in the explicit form, an interval of angles $\boldsymbol{\varphi^H_{min}}$ to $\boldsymbol{\varphi^H_{max}}$, that are considered horizontal, and an interval of angles $\boldsymbol{\varphi^V_{min}}$ to $\boldsymbol{\varphi^V_{max}}$ that are considered as vertical. The output is a set of pair of line segments that are orthogonal to each other.

function *Detect-Right-Angles* $(L_S, \varphi^H_{min}, \varphi^H_{max}, \varphi^V_{min}, \varphi^V_{max})$: **set**
$A_R \leftarrow \emptyset$
let $L_S = \{((a^1_x, a^1_y), (b^1_x, b^1_y), (\varphi^1, \rho^1)), \dots, ((a^n_x, a^n_y), (b^n_x, b^n_y), (\varphi^n, \rho^n))\}$
for $k_1 = 1,2, \dots, n$ **do**
  for $k_2 = 1,2, \dots, n$ **do**
    if $k_1 \neq k_2$ **then**
      if $\varphi^H_{min} \leq \varphi^{k_1} \leq \varphi^H_{max}$ **and** $\varphi^V_{min} \leq \varphi^{k_2} \leq \varphi^V_{max}$ **then**
        if the line segment given by points $(a^{k_1}_x, a^{k_1}_y), (b^{k_1}_x, b^{k_1}_y)$ intersects with the line segment $(a^{k_1}_x, a^{k_1}_y), (b^{k_1}_x, b^{k_1}_y)$ **then**
          $A_R \leftarrow A_R \cup \{(k_1, k_2)\}$
return A_R

Table 2. Several characteristics derived from straight lines.

Characteristic	Calculation
Length of longest segment $d_{max}^{L_S}$	$d_{max}^{L_S} = \max\{d_k \mid k = 1,2,\dots,n\}$, where $d_k = \sqrt{(a_x^k - b_x^k)^2 + (a_y^k - b_y^k)^2}$
Average segment length $d_{\mu}^{L_S}$	$d_{\mu}^{L_S} = \frac{1}{n}\sum_{k=1}^{n} d_k$
Length variance $d_{\sigma}^{L_S}$	$d_{\sigma}^{L_S} = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(d_k - d_{\mu}^{L_S})^2}$
Ratio of short segments $r_{short}^{L_S}$	$r_{short}^{L_S} = \frac{1}{n}\left\lvert\{d_k \mid 0 < d_k \leq \frac{1}{3}d_{max}^{L_S}\}\right\rvert$
Ratio of medium segments $r_{mid}^{L_S}$	$r_{mid}^{L_S} = \frac{1}{n}\left\lvert\{d_k \mid \frac{1}{3}d_{max}^{L_S} < d_k \leq \frac{2}{3}d_{max}^{L_S}\}\right\rvert$
Ratio of long segments $r_{long}^{L_S}$	$r_{long}^{L_S} = \frac{1}{n}\left\lvert\{d_k \mid \frac{2}{3}d_{max}^{L_S} < d_k \leq d_{max}^{L_S}\}\right\rvert$

5 Experimental Evaluation

The classification method based on described attributes and the decision tree has been implemented in the Java language. The training sets used for creating decision trees contained 350 manually annotated images while there were at least 40 images for each classification class (the annotation was made by the second author). All the experimental data are available at the web to allow further comparative research: http://ktiml.mff.cuni.cz/~surynek/research/micai2011.

Table 3. Accuracy on the class ***photography*** **(P)**.

	Image count	**Correctly classified**	**Accuracy**
Training	155	154	99.35%
Set A	297	243	81.82%
Set B	405	300	74.07%

Two sets of testing images called *Set A* and *Set B* can be found at the same web. These sets of images have been used for the experimental evaluation presented in this section. The Set A consists of 297 images and the Set B consists of 405 images. Each set of testing images was collected by a different user (annotation with respect to classification classes was made by the second author again).

Table 4. Accuracy on the class ***artistic images*** **(A)**.

	Image count	**Correctly classified**	**Accuracy**
Training	104	104	100.00%
Set A	297	251	84.51%
Set B	405	331	81.73%

In the experimental evaluation, we concentrated on the accuracy of the method with respect to the classification classes P, A, B, L, M (as it is given in Section 2.1). At the same time we investigated what attributes are the most important for individual

classes. Fortunately, the importance of attributes can be easily measured by their location in the decision tree – except generalization ability this was another main reason for using decision trees. The most important attributes are located closer to the root of the tree.

Results for the class **photography (P)** are shown in Table 3. The most important attributes were: ratio of colors from the color palette, the total number of colors, the number of local maxima in the histogram, and the variance around local maxima in the histogram. It is also characteristic that the number of colors is not that high for photographs (typically, rendered images contain substantially more colors).

Table 5. Accuracy on the class ***buildings*** (**B**).

	Image count	**Correctly classified**	**Accuracy**
Training	104	104	100.00%
Set A	297	232	78.11%
Set B	405	350	86.42%

Results for the class of **artistic images (A)** are shown in the Table 4. The most important characteristics are: the number of local maxima in the histogram, the variance around local maxima in the histogram, and the total number of colors. The limited number of colors that cover large areas of the image is typical for this class.

Table 6. Accuracy on the class ***landscape*** (**L**).

	Image count	**Correctly classified**	**Accuracy**
Training	90	89	98,89%
Set A	297	232	84.51%
Set B	405	350	81.73%

Classification results for the class **buildings (B)** are shown in Table 5. As it was expected, the most important attributes are: the number of right angles, the length of the longest line segment, and the ratio of long line segments.

Table 7. Accuracy on the class ***macro objects*** (**M**).

	Image count	**Correctly classified**	**Accuracy**
Training	118	118	100,00%
Set A	297	259	87.20%
Set B	405	295	72.84%

Results regarding the classification class **landscapes (L)** are shown in Table 6. Here, the most important characteristics are: the ratio of blue colors, the ratio of high-keys, the contrast of the image, and the ratio of low-keys. We can conclude that contrast plays the important role according to the expectation with respect to landscapes – natural objects exhibit high complexity which generates high contrast when photographed. The importance of blue colors can be explained by the fact, that landscape images often depict the blue sky as well.

Finally, results for the classification class of **macro objects (M)** are shown in Table 7. The most important attributes are: the area covered by non-contrast parts of the image, the ratio of high, mid, and low keys. In accordance with the expectation, the local contrast was the most important attribute.

5.1 Comparative Evaluation and Related Works

A work most related to our study is *MUFIN* – Multi-feature Indexing Network [16, 17]. It can be regarded as an image classification method that can be used for searching an image according to a keyword or according to another image. The former type of search is based on simple annotation of images using a set of keywords; the latter search is based on the similarity of images. The similarity search uses attributes such as colors and shapes depicted on the image to search for similar images.

The major difference from our work is that we are trying to generalize from a training set of images. Simply said, if we ask our method to find images that are classified as building we are likely to obtain images of various buildings which appearance was generalized from the training set. If we do the same with the *MUFIN* similarity search we will obtain images of buildings of the same shape and color as that in the reference image.Another related project is *CoPhIR* - Content-based Photo Image Retrieval Test-Collection [1] which provides a test suite for image classification. For capacity reasons we did not used this collection for experimental evaluation and used our own test suite instead.

Notice that image search provided by commercial web search engines is based on annotation of images by keywords while the search is not done over images but merely over the keywords. This represents the most primitive approach of image classification/search as there is no automation of the process.

6 Conclusions and Future Work

We proposed a relatively successful method for classification of bitmap images into selected classification classes. The classification classes are determined by the description in the natural language. The classification method itself is based on attribute extraction from the image which is subsequently processed by the decision tree.

The accuracy of the current implementation of the method ranges between 75% and 85%. We have also provided a careful analysis of the classification task for bitmap images and we described key techniques for extraction of complex attributes such as occurrence of right angles. Such complex attributes turned out to be important for distinguishing classification classes such as buildings.

We would like to emphasize that the current set of attributes and classification classes serves as the justification of the concept of using decision tree and attribute extraction for image classification. The current state does not claim to be final in any way. This attitude was kept in mind during the initial design of the concept which we therefore tried to make as extensible as possible with respect to increasing classification accuracy and extending the set of classification classes. It is planned for future work to propose more complex attributes that can be used for classification of diffi-

cult characteristics of images such as images depicting people or industrial products such as cars, planes etc. The interesting problem for future work is also how to allow the user to define her/his own classification class.

Acknowledgements. We would like to gratefully thank the Czech Science Foundation and The Ministry of Education, Youth and Sports, Czech Republic for the financial support of this work (contracts 201/09/P318 and MSM 0021620838 respectively).

References

1. Bolettieri, P., Esuli, A., Falchi, F., Lucchese, C., Perego, R., Piccioli, T., Rabitti, F.: *CoPhIR: a Test Collection for Content-Based Image Retrieval.* CoRR abs/0905.4627, http://cophir.isti.cnr.it/, [accessed June, 2011], ISTI CNR, (2009)
2. Foley, J. D., van Dam, A., Feiner, S. K., Hughes, J. F.: *Computer Graphics: Principles and Practice in C.* Addison-Wesley Professional, (1995).
3. Duda, R. O., Hart, P. E.: *Use of the Hough Transformation to Detect Lines and Curves in Pictures.* Communications of the ACM, ACM Press, (1972).
4. Lukšová, I.: *Klasifikace bitmapových obrázků (Classification of Bitmap Images).* Bachelor thesis, Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic, (2010).
5. Marr, D., Hildreth, E. C.: *Theory of edge detection.* Proceedings of the Royal Society of London, Series B, Volume 207 (1167), pp. 187–217, The Royal Society, (1980).
6. Mitchell, T. M.: *Machine Learning.* McGraw Hill, (1997).
7. Peli, E.: *Contrast in Complex Images.* Journal of the Optical Society of America A: Optics, Image Science, and Vision, Volume 7 (10), pp. 2032-2040, OSA, (1990).
8. Quinlan, J. R.: *Induction of Decision Trees.* Machine Learning, Volume 1, pp. 81-106, Springer, (1986).
9. Quinlan, J. R.: *C4.5: Programs for Machine Learning.* Morgan Kaufmann, (1993).
10. Rabbani, T., van den Heuvel, F.: *Efficient Hough transform for automatic detection of cylinders in point clouds.* ISPRS Workshop Laser scanning 2005, pp. 60-65, Institute of Photogrammetry and Remote Sensing, (2005).
11. Rokach, L., Maimon, O.: *Top-down induction of decision trees classifiers: a survey.* IEEE Transactions on Systems, Man, and Cybernetics, Part C 35, pp. 476–487, IEEE Press, (2005).
12. Russel, S., Norvig, P.: *Artificial Intelligence – A modern approach.* Prentice Hall, (2003).
13. Shapiro, L. G., Stockman, G. C.: *Computer Vision.* Prentice Hall, (2001).
14. Shirley, P., Ashikhmin M., Marschner, S.: *Fundamentals of Computer Graphics.* A.K. Peters, (2009).
15. Yahoo! Inc.: *flickr from Yahoo! - almost certainly the best online photo management and sharing application in the world.* Commercial web site, http://www.flickr.com/, [accessed June, 2011].
16. Zezula, P., Amato, G., Dohnal, V., Batko, M.: *Similarity Search - The Metric Space Approach.* Advances in Database Systems, Springer, (2006).
17. Zezula, P., Amato, G., Dohnal, V., Batko, M.: *MUFIN – Multi-feature Indexing Network / Image Search.* Project web site, http://mufin.fi.muni.cz/imgsearch/, [accessed June, 2011], Masaryk University, Czech Republic, (2008).
18. Zhai, L., Dong, S., Ma, H.: *Recent Methods and Applications on Image Edge Detection.* International Workshop on Education Technology and Training and International Workshop on Geo-Science and Remote Sensing, pp. 332-335, IEEE Press, (2008).