Edge information based object classification for NAO robots

Karl Tarvas1, Anastasia Bolotnikova1 and Gholamreza Anbarjafari1*

Abstract: This paper presents a research regarding the development of a computationally cheap and reliable edge information based object detection and classification system for use on the NAO humanoid robots. The work covers ground detection, edge detection, edge clustering, and cluster classification, the latter task being equivalent to object recognition. In this work, a new geometric model for ground detection, a joint edge model using two edge detectors in unison for improved edge detection, and a hybrid edge clustering model have been proposed which can be implemented on NAO robots. Also, a classification model is outlined along with example classifiers and used values.

Subjects: Computer Science (General); Digital Signal Processing; Image Processing

Keywords: edge detection; clustering; random forest; object recognition; machine vision

ABOUT THE AUTHORS
Karl Tarval received his BSc in Computer Engineering from University of Tartu in 2016. He is currently a senior software developer.

Anastasia Bolotnikova is a master's student at faculty of Computer Science in University of Tartu where she obtained her BSc as well. She is a member of iCV Research group, Institute of Technology at University of Tartu since 2014. She joined the RoboCop team of University of Tartu as team Philosopher, in fall 2014. Currently she is working in the field of image processing developing the real-time self-localization algorithm for Nao robots. Her BSc thesis work was granted the second best thesis work in the faculty of Computer Science.

Gholamreza Anbarjafari received his PhD from the Department of Electrical and Electronic Engineering at Eastern Mediterranean University (EMU) in 2010. He has been working in the field of image processing and is currently focusing in many research works related to multimodal emotion recognition, image illumination enhancement, super resolution, image compression, watermarking, visualization and 3D modeling, and computer vision for robotics. He is currently head of iCV Research Group and is working as an associate professor in Institute of Technology at University of Tartu. He is an IEEE senior member and the vice chair of Signal Processing Chair of IEEE Estonian section. He also holds Estonian Research Council’s grant (PUT).

PUBLIC INTEREST STATEMENT
This paper presents a research regarding the development of a computationally cheap and reliable edge information based object detection and classification system for use on the NAO humanoid robots. This work will introduce how a NAO robot can start to recognize the environment for better localization with minimum dependency on color information. The work covers ground detection, edge detection, edge clustering, and cluster classification, the latter task being equivalent to object recognition. In this work a new geometric model for ground detection, a joint edge model using two edge detectors in unison for improved edge detection, and a hybrid edge clustering model have been proposed which can be implemented on NAO robots. Also, a classification model is outlined along with example classifiers and used values.
1. Introduction

The NAO humanoid robot is a programmable robot developed by Aldebaran Robotics. The robot is widely used both in academia and in the private sector for research and other educational purposes (Unveiling of NAO Evolution, 2014). The NAO is currently the standard robot used for the Robot Soccer World Cup, RoboCup for short, in which teams from across the world compete in robot soccer and other events annually (RoboCup Standard Platform League rules, 2014).

The NAO is 58cm tall, weighs 4.3kg, and has a total 25 degrees of freedom in its joint control. All of the robot's software is run on a single Intel Atom 1.6 GHz processor, making multitasking and complex procedures a challenge (Aldebaran NAO Documentation, 2015). All processing power must be shared between the robot's custom Linux-based OS NAOqi and different modules which handle moving, multiple sensors, communication etc. As the hardware platform is fixed and no modifications are allowed, all teams compete on the same basis (RoboCup Standard Platform League rules, 2015). The NAO's main source of information is vision, provided by two cameras, each with a maximum resolution of $1,280 \times 720$ px. Additionally, the robot has infrared sensors, tactile sensors, pressure sensors, and other systems, all of which will not be covered further herein.

The RoboCup hosts numerous different competitions for robots, only the Standard Platform League (SPL) soccer competition scenario will be addressed from here on out. The competition features two teams playing on opposite sides of a green field analogous to a scaled down version of a regular soccer field. During the competition, the robots must operate autonomously both to cooperate as a team and to play as an individual player (RoboCup Standard Platform League rules, 2014). Interpreting information provided by the cameras quickly and accurately is a critical prerequisite for succeeding in that task.

In earlier years, the RoboCup competition field consisted of components with unique color characteristics: yellow goal posts, orange soccer ball, green field area, etc. As the complexity of the participating teams' software has improved, the field setup has been modified to better match that of an actual soccer field: the goal posts are now white and the ball is a black and white truncated icosahedron (RoboCup Standard Platform League rules, 2014), both shown in Figure 1.

Since numerous objects of interest, namely goal posts, robots, ball, and field lines, are now all dominantly white, an approach based solely on color information is insufficient for a reliable model as demonstrated in Bolotnikova (2015). The main contribution of this work is lying on using the following three main design principles:
• Computation speed—information must be provided rapidly to enable the robot to make ade-
quate decisions during the game;
• Conservative use of resources—the NAO’s single Intel Atom processor is shared by all its sys-
tems (Aldebaran NAO Documentation, 2015);
• Universality—the module must work reliably regardless of fluctuations in lighting and noise.

To successfully implement the module, some important topic, namely, ground detection, edge
detection, edge clustering, and cluster classification, will be discussed within this work. Each section
will be analyzed from the perspective of the above main principles. In this work for more robust
classification new method is introduced which is incorporating detection of edge using random for-
est and canny edge detector.

2. Ground detection

In order to detect and classify different objects properly, identifying the area of the playing field cur-
cently in view is a crucial prerequisite. Determining the field’s area divides all items of interest into
two categories (“in the field” or “not in the field”) and gives valuable information regarding the
robot’s location on the playing area. Using the histogram normalization technique (Sridharan &
Stone, 2009; Anbarjafari, Jafari, Jahromi, Ozcinar, & Demirel, 2015) and initial mean values proposed
in Bolotnikova (2015) for similar purposes, the green playing field area can be easily detected by set-
ing a threshold value. This process, however, can leave many areas where the view may be
obstructed by other robots excluded, as demonstrated in Figure 2, where both the ball and the
nearby robot are considered “not in the field” by the naive thresholding approach.

To bypass these occlusions, a simple geometric approach is proposed as follows: given an input
image directly from the camera \(\Psi_{\text{raw}}\), the image is sized down to make all further operations com-
putationally cheaper. The image is resized by a heuristically determined factor of \(\frac{1}{9}\), i.e. both the
width and the height of the input image will be a third of their original size. From here on out, \(\Psi\) shall
refer to the resized input frame.

\[
\Psi_{\text{raw}} \xrightarrow{\frac{1}{9}} \Psi
\]

(1)

Basic color thresholding is applied to \(\Psi\) based on values from Bolotnikova (2015), yielding a binary
array \(\Psi_B\). All areas of set bits under a heuristically determined surface area are unset, reducing the
amount of noise present in \(\Psi_B\). In practice, a cheap erode and dilate is used with a rectangular
morph with a heuristically determined side length of 10px, a sample is shown in Figure 3. The lowest
corner points for the ground area are found:

\[
\{L_l, L_r\} \in R
\]

(2)

where \(R \subseteq \Psi_B\) is the detected ground region and subscripts \(l\) and \(r\) refer to left and right, respectively.
Each lowest corner is corresponding to a highest point roughly above it, i.e. there exist \(A_l\) and \(A_r\),
which satisfy:
where \( \varepsilon_{\text{snap}} \) is chosen heuristically. In practice, 
\[
\varepsilon_{\text{snap}} = 20 \text{ px}
\]  

Note that:
\[
\{ A_l, A_r \} \in R
\]  

Provided all work is conducted in the OpenCV standard Cartesian coordinate system (Laganière, 2011) demonstrated in Figure 3, for each point in \( R \), a weight \( \omega \) is calculated by:
\[
\omega^{(x,y)} = \begin{cases} \frac{\sqrt{y}}{\|\Psi(W-1,0)-\Psi(x,y)\|} & \text{if } x < \frac{L_l^x + L_r^x}{2} \\ \frac{\sqrt{y}}{\|\Psi(0,0)-\Psi(x,y)\|} & \text{if } x \geq \frac{L_l^x + L_r^x}{2} \end{cases} \quad (x,y) \in R
\]  

where \( W \) is the width of \( \Psi \) in pixels. For each half portion of the region between the bottom corners
\[
S_l = \max_{} \omega^{(x,y)}, \quad x < \frac{L_l^x + L_r^x}{2}
\]
\[
S_r = \max_{} \omega^{(x,y)}, \quad x \geq \frac{L_l^x + L_r^x}{2}
\]  

are then defined (Figures 4 and 5). While they overlap with previously found points in many generic scenarios, as can be seen in Figure 6.
Additionally, for each half portion of the same region, minima are selected by

\[ M_l(x, y) = \min_y R, \quad x < \frac{L_l + L_r}{2} \]

\[ M_r(x, y) = \min_y R, \quad x \geq \frac{L_l + L_r}{2} \]

Finally, all points defined above are snapped to the closest edges of \( \Psi_B \) within a small threshold \( \varepsilon_{\text{snap}} \) defined prior. This approach yields an eight-vertex polygon which is then padded to ensure all objects of interest that should be classified as “in the field”, are classified as such reliably. The resulting polygon closely approximates the ground area regardless of occlusions and viewport orientation, as can be seen in Figure 7. The proposed method is a computationally cheap way (\( O(N) \), where \( N \) is the size of \( \Psi \)) to closely estimate the position of the playing field in the current video frame, i.e. to find a subset \( F \) representing the field from the input image \( \Psi \):

\[ F \subseteq \Psi \]

The approach can be sensitive to large areas of noise of matched color in areas outside the field. If necessary, additional filtering can be performed, but current testing has shown no need for further processing.
3. Edge detection

3.1. Motivation

An edge $E \subseteq \Psi$ is a part of an image where significant variations in color intensity or brightness occur (Canny, 1986; Gomes, 2011; Oskoei & Hu, 2010). Discontinuities in said properties generally correspond to changes in either depth, surface orientation, material properties, or scene illumination (Oskoei & Hu, 2010), and as such offer valuable information regarding the contents of the image.

Edge detection refers to a collection of different algorithms which aim to identify the edges in an input image (Oskoei & Hu, 2010). Classical edge detection algorithms can broadly be divided into two categories: first derivative based, also known as Gradient, and second derivative based, also known as Laplacian (Šimec, 2014; Yitzhaky & Peli, 2003; Ziou & Tabbone, 1998). First-derivative-based methods look for local extrema in the first derivative of the input function, second-derivative-based methods look for zero crossings in the second derivative of the input function.

Classical edge detection algorithms convolve an input image with a two-dimensional operator $O$ characteristic to that specific detector, yielding a grayscale response where edges are distinctively shown with either maxima or minima (Gomes, 2011):

$$\Psi \ast O = \Psi_g$$

(11)

Giving $O$ different properties affects a detector’s sensitivity to fine detail (and as such, noise), different types of edges (thick or thin, consistent or inconsistent etc.) and different edge orientations (Sharifi, Fathy, & Mahmoudi, 2002). The computational complexity of the filter is also directly related to both the size and computation cost of the operator.

Gradient-based methods use two different kernels, either $O_x$ and $O_y$ one for horizontally inclined edges and the other for vertical edges, or $O_{\alpha}$ and $O_{\beta}$, one for each diagonal direction (Oskoei & Hu, 2010). Using two kernels yields two separate grayscale responses:

$$\Psi \ast O_x = \Psi_{G_x}$$
$$\Psi \ast O_y = \Psi_{G_y}$$

(12)

A very commonly used (Sharifi, et al., 2002) example of a gradient-based approach is the Sobel operator, which uses
as its kernels (Duda & Hart, 1973). Given the above, the general gradient magnitude can be obtained by

$$|\Psi| = \sqrt{\frac{\partial^2 \Psi}{\partial x^2} + \frac{\partial^2 \Psi}{\partial y^2}}$$

commonly approximated instead by

$$|\Psi| \approx \left| \frac{\partial \Psi}{\partial x} \right| + \left| \frac{\partial \Psi}{\partial y} \right|$$

as the latter is much faster to compute (Gomes, 2011). Using separate kernels also makes edge directions easily computable from the responses, e.g. given responses from the aforementioned kernels (Ziou & Tabbone, 1998):

$$\Psi_\theta = \tan^{-1}\left( \frac{\Psi_y}{\Psi_x} \right)$$

Other first-derivative-based methods compute gradient magnitude and edge direction in an analogous manner, with possible constants depending on kernel properties (Gomes, 2011).

Laplacian-based methods rely on the Laplace operator $\Delta$, which is a differential operator given by the divergence of a function’s gradient in Euclidean space (Vinogradov & Hazewinkel, 2001), given in two dimensions as (Ziou & Tabbone, 1998):

$$\Delta \psi(x, y) = \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2}, \ (x, y) \in \Psi$$

For edge detection, approximate two-dimensional convolution kernels are used instead as the input space is discrete, e.g. Games (2011):

$$O_L = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

All of the above methods are highly sensitive to noise and are generally used with an additional smoothing step, commonly convolving with a discrete approximation of a Gaussian filter. Since convolution is associative, smoothing can be applied to $O$ prior to convolving, instead of applying it directly to $\Psi$. This makes computation cheaper, as for all common cases the size of $O$ is considerably smaller than the size of $\Psi$ (Gomes, 2011).

### 3.2. Canny

The edge detection approach proposed in Canny (1986), commonly referred to as “Canny edge detector” (Gomes, 2011), is a multiple stage algorithm based on optimizing functionals, i.e. functions mapping an input vector to a scalar, for detection (identifying edges), localization (locating edges), and singularity (identifying each edge at most once) on the operator’s impulse response (Canny, 1986). Prior to convolving, the input is smoothed by an approximation of a Gaussian filter. The detector uses four kernels, $O_{x}, O_{y}, O_{D_1}$, and $O_{D_2}$, which respond maximally to horizontal, vertical, upward diagonal, and downward diagonal edges, respectively, with specific kernel values depending on a
given implementation (Oskoei & Hu, 2010). For each operator’s response, non-maximum supression is applied, resulting in thinner, well-defined edge candidates, after which the responses are merged and hysteresis is applied, resulting in binary edges (Canny, 1986). Hysteresis in edge detection means tracking all edge candidates using two thresholds, a lower one and a higher one, \( \epsilon_{\text{lower}} \) and \( \epsilon_{\text{higher}} \) respectively.

\[
\begin{align*}
\epsilon_{\text{lower}} & \in \mathbb{R} \\
\epsilon_{\text{higher}} & \in \mathbb{R}
\end{align*}
\]

(19)

All points with brightness above \( \epsilon_{\text{lower}} \) that can be connected to a point with brightness above \( \epsilon_{\text{higher}} \) without any intermediate point having a value below \( \epsilon_{\text{lower}} \) are set to full brightness, all others are suppressed (Gomes, 2011). This means hysteresis can be used to map a grayscale input to a binary output:

\[ \Pi_{\text{hyster}} : \Psi_G \rightarrow \Psi_B \]

(20)

In practice, a heuristically chosen combination of the mean value and the standard deviation of the grayscale input frame are used to find suitable threshold values:

\[
\begin{align*}
\epsilon_{\text{lower}} & = \Psi_G - \frac{\sigma(\Psi_G)}{2} \\
\epsilon_{\text{lower}} & = \Psi_G + \frac{\sigma(\Psi_G)}{2}
\end{align*}
\]

(21)

The approach yields accurate, consistent binary edges, demonstrated in Figure 8. The result is improved further when histogram equalization has been previously applied to the grayscale input (Rizon, 2006).

Canny’s algorithm is the most commonly used edge detection algorithm due to its reliability, low complexity and availability (Sharifi et al., 2002). However, the algorithm is highly sensitive to fine detail, oftentimes more sensitive than required, and scenario-specific parametrization is a prerequisite for good results (Oskoei & Hu, 2010). The same problem applies to the current setting, as demonstrated in figure 8: while sufficient detail is obtained in the playing area, objects outside of the playing area can create a lot of unrelated information which will still need to be processed. An efficient solution to the issue is proposed later in Section 3.4.

3.3. Random forests

Random forests is a generic supervised machine learning algorithm that can be used for classification, regression, and other similar tasks. The algorithm outlined in Breiman (2001) consists of independently training a large group of decision trees, then passing the input data to each tree individually which then collectively vote to identify the best candidate output label according to an ensemble model. A crucial part of the system is recursively training each tree so that the remaining data is split at each new node to achieve a large identifying information difference between the branches. Geurts, Ernst, and Wehenkel (2006) demonstrates that up to a certain limit, introducing more randomness at node level yields higher accuracy forests and therefore perfect splits are actually detrimental to overall ensemble performance and as such, undesired. Random forests can be trained and stored beforehand which make them a good candidate for systems with reasonable amounts of

Figure 8. From left to right: video frame \( \Psi \), binary frame \( \Psi_B \) from Canny’s edge detector, manually annotated \( \Psi \), where green marks edges that offer valuable information and red marks all other edges.
storage but no strong computing power, such as the NAO. Once trained, the importance of each input variable can be deducted from the model with reasonable ease, giving valuable insights for further configuration (Breiman, 2001). The algorithm is both fast (Breiman, 2001; Dollár & Zitnick, 2013; Roy & Larocque, 2012) and, given a reasonably large training set, very accurate (Geurts et al., 2006).

Dollár and Zitnick (2013) proposes extending random forests to general structured output spaces in such a manner that an input image patch \( P \subseteq \Psi \) can be mapped to a corresponding label, creating a novel type of edge detector that inherits the previously mentioned benefits of generic random forests. The central issue for the approach is comparing similarity during the training process, which is not well defined over the output space. To bypass the problem, an intermediate mapping \( \Pi_{\text{simil.}} \) from the input space \( \Psi \) to an Euclidean space \( Z \) is used,

\[
\Pi_{\text{simil.}}: \Psi \rightarrow Z
\]  

where comparing similarity can simply be done by comparing Euclidean distance. In order to avoid the issues outlined by Geurts et al. (2006), a new mapping is randomly generated for each tree to ensure sufficient levels of deviation from the norm at the node level. To reduce the amount of noise generated by the randomness component, each point is oversampled, i.e. there exist two different patches \( P_1 \) and \( P_2 \) such that

\[
\exists \{P_1, P_2\}: P_1 \cap P_2 \neq \emptyset \tag{23}
\]

The results are averaged across patches which can potentially lead to a general loss of accuracy. To counter the issue (Dollár & Zitnick, 2013), runs the algorithm at multiple scales, labeling the input at the original, half and double the resolution, and then averaging the results after resizing each back to original input dimensions. Based on practical application in the industry, the original authors proceed to outline further scenario-specific optimizations in Dollár and Zitnick (2015).

An edge detector based on random forests trained with the properties proposed by Dollár and Zitnick (2013) has characteristically soft edges as shown in Figure 9. As a result of oversampling, the detector inherently discriminates against noise and detects the dominant features in an image which are more likely to hold interesting information (Dollár and Zitnick, 2013). While this works well for general edge detection cases, in the given setting, crucial information may be lost in numerous scenarios as demonstrated in Figure 9. This issue is addressed in a later Section 3.4.

3.4. Merged edge model

As covered prior, both Canny’s edge detector and the random forests edge detector have failure cases in which either too much noise is detected or too much detail omitted, respectively, shown in Figures 8 and 9. As the problem can be isolated to the area considered “not in the field” for the former algorithm and to the area considered “in the field” for the latter, a simple combined model is proposed. Canny’s algorithm is used to detect edges from the area considered “in the field”:
and the random forests edge detector is used to detect edges elsewhere:

\[
\Psi_{\text{Rand.For.}} = \Psi \setminus F
\]  

(25)

To avoid breaking consistent edges, both \(\Psi_{\text{Canny}}\) and \(\Psi_{\text{Rand.For.}}\) are padded with a small value \(\epsilon_{\text{overlap}}\) to create overlap, in practice

\[
\epsilon_{\text{overlap}} = 3 \text{ px}
\]  

(26)

where \(\epsilon_{\text{overlap}}\) is chosen heuristically. Using too large values for \(\epsilon_{\text{overlap}}\) creates unwanted noise while using too small values yields edges that are disconnected at the boundary of the two areas. As such, choosing an optimal value is critical.

The edges from the random forest edge detector are binarized using hysteresis with heuristic parameters

\[
\epsilon_{\text{higher}} = \frac{1}{2}, \quad \epsilon_{\text{lower}} = \frac{1}{6}
\]  

(27)

assuming all values fall within \([0, 1]\). This approach has suboptimal accuracy as many edges are not singular, but it is fast and sufficiently accurate with the proposed classification model, outlined in the following sections.

The results from the two detectors, \(\Psi_{\text{B_Canny}}\) and \(\Psi_{\text{B_Rand.For.}}\), are then joined using bitwise or, annotated with \(\text{var}\), resulting in a binary output, demonstrated in Figure 10:

\[
\Psi_{\text{B}} = \Psi_{\text{B_Canny}} \mid \Psi_{\text{B_Rand.For.}}
\]  

(28)

4. Edge clustering

Each point \(Q \in E\) in every edge \(E \subseteq \Psi_{\text{G}}\) has a corresponding direction \(\theta_{Q}\) that is equal to the gradient direction at that point, i.e.

\[
\theta_{Q(x,y)} = \Psi_{\text{G}}(x,y), \quad (x,y) \in \Psi_{\text{G}}
\]  

(29)

All directions are quantized into four categories analogous to the approach outlined in Canny (1986) and a label \(L\) is associated with every edge point using

\[
L(Q) = \left[ \frac{4\theta_{Q}}{\pi} - \frac{1}{2} \right] \mod 4 + 1
\]  

(30)

visualized in Figure 11.

Similar to the approach proposed in Zitnick and Dollar (2014), edges are then grouped by joining all eight-connected points, except only edges with an identical label are joined, forming a cluster.

Figure 10. From left to right: video frame \(\Psi\), grayscale frame \(\Psi_{\text{G}}\) resulting from combining the results of Canny’s edge detector and the random forests edge detector without hysteresis, binary \(\Psi_{\text{B}}\) after hysteresis is applied.
demonstrated in Figure 12. Using the relative direction difference proposed in Zitnick and Dollár (2014) without quantized labels was tested, but proved less reliable in the given setting. Every point may be a member of at most one cluster, i.e. for any two clusters $C_1$ and $C_2$:

$$C \subseteq E$$

Figure 11. Quantized direction labels.

Figure 12. Sample patch of quantized directions, red marks possible cluster creation.

$$\exists Q: Q \in C_1, Q \in C_2$$

Grouping is done recursively, proceeding along the horizontal axis of $\Psi_G$ at first, as the memory addresses are sequential, and then vertically, as is the industry standard practice (Laganière, 2011). Clusters with mass under a small heuristically determined threshold $\epsilon_{\text{mass}}$ are discarded as noise.

$$\epsilon_{\text{mass}} = 5 \text{ px}$$

5. Classification of edge clusters

5.1. General overview

Multi-variable decision models have been shown to consistently outperform holistic, single descriptor classification models (Serre, Wolf, & Poggio, 2005). Many general algorithms have been proposed for both detecting and classifying objects. Numerous commonly used approaches are unfeasible for the given setting: some approaches are licenced prohibitively (e.g. Bay, Tuytelaars, & Van Gool, 2006; Lowe, 2004), some are too general and as such computationally too expensive (e.g. Rosten &
Drummond, 2006; Rublee, Rabaud, Konolige, & Bradski, 2011). Using the proposed merged edge model with the clustering approach based on Canny (1986), Zitnick and Dollár (2014), the number of candidates both calculated and checked against can be reduced greatly. As such, a simple classification model is constructed on the general principles of Lowe (2004), Zitnick and Dollár (2014): based on the computed cluster information, each observed variable is given a relatively wide acceptance range, as opposed to a single narrowly ranged variable based model, i.e. a collection of weak classifiers is used instead of a single strong classifier. A weak classifier is a classifier that accepts a wide range of values as matching, a strong classifier accepts a narrow range, a sample comparison is shown in Figure 13. Multiple weak classifiers working in unison make noise in any input variable less relevant (Breiman, 2001).

Classifying the clusters can be done by storing a number of characteristic properties for each of them. For every cluster, two points contained in it with the longest distance between them, \( Q_1 \) and \( Q_2 \), are found, which can be done cheaply during cluster creation.

\[
\{Q_1, Q_2\} \in C
\]  

(34)

The distance between \( Q_1 \) and \( Q_2 \) is used as a cheap approximation for the running length of a cluster. Additionally, \( Q_3 \) is found which is a point directly between \( Q_1 \) and \( Q_2 \).

\[
\|Q_1 - Q_3\| = \|Q_2 - Q_3\| = \frac{||Q_1 - Q_2||}{2}
\]  

(35)

It is worth noting that \( Q_3 \) may not necessarily be a point on any edge:

\[
\exists C: Q_3 \notin E \Rightarrow Q_3 \notin C
\]  

(36)

For all three points, brightness of \( \Psi \) at the given point is stored as well as whether the point is considered “in the field” or “not in the field” by ground detection. Additionally, the parameters for a bounding rectangle are found for each cluster along with the average gradient orientation. The cluster mass based classification heuristic (where the mass of a cluster is equal to the number of unique pixels contained in it) proposed in Oskoei and Hu (2010) is also employed.

Following, two sample classifications are described without probabilistic components, constructing a probabilistic model is considered out of scope for this work. All thresholds are determined heuristically given an input image \( \Psi \), where \( \Psi \) is scaled down as outlined in “Ground detection”.

Figure 13. From left to right: a single strong classifier where a small deviation from the threshold values can mean a mismatch, a probabilistic collection of weak classifiers where noise in any input variable affects the final classification considerably less.

Notes: Red marks classification matching, the vertical axis shows ranges for different variables, horizontal axis shows different classifiers.
5.1.1. Goals
A goal can be described by either one or two goalposts that have their bottoms on the ground area and also have a connecting part at the top. Identifying goals is a three-step process: identify candidate goalposts, identify top connecting parts, and finally remove all goalpost candidates that are not connected at the top. Initial goalpost candidates are picked using constraints on the average direction of the cluster, the running length of the cluster and the properties of \( Q_1, Q_2 \) and \( Q_3 \) for that cluster, all threshold values are chosen heuristically.

\[ C_\mu, \text{the average direction for the cluster must fall within} \]

\[ C_\mu \in \left( \frac{\pi - 0.4}{2}, \frac{\pi + 0.4}{2} \right) \]  

(37)

\[ C_{\text{length}}, \text{the running length of the cluster, must be between} \]

\[ C_{\text{length}} \in (40 \text{ px}, 200 \text{ px}) \]  

(38)

\[ C_F \text{ shows how many of } \{Q_1, Q_2, Q_3\} \text{ for that specific cluster are considered “in the field”:} \]

\[ C_F = \left| \left\{ Q; Q \in \{C_{Q_1}, C_{Q_2}, C_{Q_3}\}, Q \in F \right\} \right| \]  

(39)

For initial goalpost candidates, this value must be

\[ C_F \leq 1 \]  

(40)

If all the above criteria are met, the given cluster is added to the list of initial goalpost candidates. Top connectors are selected from the remaining clusters using constraints on position and distance. The distance between either end of the given cluster, i.e. \( Q_1 \) or \( Q_2 \) for that cluster, and any goalpost candidate must be under a threshold:

\[ \epsilon_{\text{connector}} = 20 \text{ px} \]  

(41)

Additionally, the approximate center for the given cluster must be higher than the goalpost candidate’s approximate center, i.e. \( Q_3 \) for the given cluster must have a lower ordinate value than the \( Q_3 \) of the candidate goalpost, given the standard OpenCV Cartesian system (Laganière, 2011).

Finally, all goalpost candidates that do not have a matching connector are removed from the list of matching candidates. Sample detections are shown in Figures 14 and 15.

---

**Figure 14.** From left to right: original video frame \( \Psi \), half opacity \( \Psi \) with annotations.

Notes: Yellow marks detected goal posts, red marks detected top connectors.
5.1.2. Ball
A ball can be described as a collection of small clusters where both very high brightness (white) and very low brightness (black) are present nearby and the clusters are considered “in the field”. Ball clusters are identified using constraints on brightness around cluster, darkness around cluster, cluster mass, and saturation, all with heuristically chosen threshold values.

\[ C_{m}, \text{ the cluster's mass in pixels, must be under a given size} \]
\[ C_{m} < 20 \text{ px} \]  \hspace{1cm} \text{(42)}

\[ C_{\text{sat}}, \text{ saturation around the cluster's end points } Q_1 \text{ and } Q_2 \text{ in a bounding rectangle with side } r_{C_{\text{sat}}} \text{ must not change more than a given value, i.e. assuming saturation values fall in [0, 1]}: \]
\[ C_{\text{sat}} < \frac{1}{2} \]
\[ r_{C_{\text{sat}}} = 5 \text{ px} \]  \hspace{1cm} \text{(43)}

\[ C_{\text{bright}}, \text{ the highest brightness value in a bounding rectangle with a side length } r_{C_{\text{right}}} \text{ around both of the cluster's end points must be at least a given value, assuming all values fall in range } [0, 1]: \]
\[ C_{\text{bright}} > \frac{4}{10} \]
\[ r_{C_{\text{right}}} = 3 \text{ px} \]  \hspace{1cm} \text{(44)}

\[ C_{\text{dark}}, \text{ the lowest brightness value with similar configuration must be below a given value} \]
\[ C_{\text{dark}} < \frac{4}{10} \]
\[ r_{C_{\text{dark}}} = 3 \text{ px} \]  \hspace{1cm} \text{(45)}

\[ C_F, \text{ described before, must be sufficiently high} \]
\[ C_F \geq 2 \]  \hspace{1cm} \text{(46)}

These values reliably locate the ball on views where the ball is nearby, a sample detection is shown in Figure 16. The developed algorithm has been tested on 100 real-world scenarios in which the ball was placed on different positions and the NAO robot detected the ball for 98 occasions.
6. Conclusion and future work

The basis of a new edge information-based vision module for the NAO humanoid robot has been proposed and implemented. A new method of ground detection has been outlined, building on the work in Bolotnikova (2015), improving the previous approach by removing any existing occlusions. A new merged edge model has been proposed that uses the algorithms outlined in Canny (1986) and Dollár and Zitnick (2013) in unison. As a prerequisite for object classification, an edge clustering method is proposed by combining the approaches from Zitnick and Dollár (2014) and Canny (1986). Building upon it, a non-probabilistic edge information-based object detector and classifier has been implemented. Sample-classified objects have been outlined and tested, demonstrating the proposed classifier’s functionality, its strong points and places where improvements can be made. Each step has been covered in sufficient detail, explaining the design decisions. A solid foundation for further work has been laid, outlining both specific improvements to build upon as well as long-term outlooks for future research.

The current implementation is a coarse base demonstrating the viability and efficiency of the proposed approach. It is open to both model-scale optimizations covered in Hinterstoisser et al. (2012), precision improvements outlined in Dollár and Zitnick (2015) and cluster merging outlined in Zitnick and Dollár (2014). The binarization of the random forest edge detector can be improved using the approaches outlined in Canny (1986). The currently used random forest edge detection model is trained with parameters similar to those outlined in Breiman (2001) on the general BSDS500 dataset (Arbelaez, Maire, Fowlkes, & Malik, 2011). Using a different dataset, using a specifically constructed subset of the one currently used or tuning the forest parameters may provide a final model that is smaller and faster to operate on. The implemented classifier is non-probabilistic and does not fully leverage all of the data available. However, the proposed classification model is a perfect candidate for machine learning.

Funding
This work has been partially supported by Estonian Information Technology Foundation, Skype Technologies, IdevLab of University of Tartu, and Estonian Research Council [grant number PUT638].

Author details
Karl Tarvas1
E-mail: karl.tarvas@ut.ee
Anastasia Bolotnikova1
E-mail: nana@icv.tut.ut.ee
Gholamreza Anbarjafari1
E-mail: shb@icv.tut.ut.ee
ORCID ID: http://orcid.org/0000-0001-8460-5717
1 ICV Research Group, Institute of Technology, University of Tartu, Tartu 50411 Estonia.

Citation information
Cite this article as: Edge information based object classification for NAO robots, Karl Tarvas, Anastasia Bolotnikova & Gholamreza Anbarjafari, Cogent Engineering (2016), 3: 1262571.

References
Aldebaran NAO Documentation. (2015). Retrieved May 12, 2016, from http://doc.aldebaran.com/1-14/family/nao_h25/index_h25.html
Anbarjafari, G., Jafari, A., Jahromi, M. N. S., Ozcinar, C., & Demirel, H. (2015). Image illumination enhancement with an objective no-reference measure of illumination assessment based on Gaussian distribution mapping. Engineering Science and Technology, An International Journal, 18, 696–703.
Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. (2011). Contour detection and hierarchical image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33, 898–916.
Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Computer vision—ECCV (pp. 404–417). Graz: Springer.
Bolotnikova, A. (2015). Melioration of color calibration, goal detection and self-localization systems of NAO humanoid robots (Thesis). University of Tartu, Tartu.
Breiman, L. (2001). Random forests. Machine Learning, 45, 5–32.
Canny, J. (1986). A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 6, 679–698.
Dollár, P., & Zitnick, L. (2013). Structured forests for fast edge detection. Proceedings of the IEEE International Conference on Computer Vision (pp. 1841–1848). Sydney.

Dollár, P., & Zitnick, L. (2015). Fast edge detection using structured forests. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37, 1558–1570.

Duda, R. O., & Hart, P. E. (1973).

Dollár, P., & Zitnick, L. (2015). Fast edge detection using structured forests.

Dollár, P., & Zitnick, L. (2013). Structured forests for fast edge detection.

Rosten, E., & Drummond, T. (2006). Computer Vision-ECCV. Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 2564–2571). Barcelona.

Serre, T., Wolf, L., & Poggio, T. (2005). Object recognition with features inspired by visual cortex. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005) (pp. 994–1000). New York, NY.

Sharifi, M., Fathy, M., & Mahmoudi, M. T. (2002). A classified and comparative study of edge detection algorithms. Proceedings International Conference on Information Technology: Coding and Computing (pp. 117–120). Las Vegas, NV.

Šimec, A. (2014). Ocenenjevane relativine zmogljivosti algoritmov za odkrivanje robov [Assessing relative performance of edge detection algorithms] (Ph.D. thesis), Univerza v Ljubljani, Ljubljana.

Sridharan, M., & Stone, P. (2009). Color learning and illumination invariance on mobile robots: A survey. Robotics and Autonomous Systems, 57, 629–644.

Unveiling of NAO Evolution: A stronger robot and a more comprehensive operating system. (2014). Retrieved May 19, 2016, from https://www.ald.softbankrobotics.com/en/press/press-releases/unveiling-of-nao-evolution-a-stronger-robot-and-a-more-comprehensive-operating.

Vinogradov, I.M., & Hazewinkel, M. (2001). Encyclopaedia of mathematics. New York, NY: Kluwer Academic.

Yitzhaky, Y., & Peli, E. (2003). A method for objective edge detection evaluation and detector parameter selection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25, 1027–1033.

Zitnick, C. L., & Dollár, P. (2014). Computer Vision-ECCV. Springer, 2014, 391–405.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. 2011 International conference on computer vision (ICCV) (pp. 2564–2571). Barcelona.

Roy, M.-H., & Larocque, D. (2012). Robustness of random forests for regression. Journal of Nonparametric Statistics, 24, 993–1006.