University of Tartu
Faculty of Science and Technology
Institute of Technology
Computer Engineering Curriculum

Karl Tarvas

Edge information based object detection and classification

Bachelor's thesis (12 ECTS)

Supervisor: Assoc. Prof. Gholamreza Anbarjafari
Pejman Rasti, MSc

Tartu 2016

Ääreinformatsioonil põhinev esemete tuvastamine ja klassifitseerimine

Lühikokkuvõte:

Käesolevas bakalaureusetöös esitletakse arvutuslikult odava ja töökindla, ääreinformatsioonil põhineva esemete tuvastamise ja klassifitseerimise süsteemi arendamist, mis on rakendatav NAO humanoidrobotitel. Töös kajastatakse järgnevaid teemasid: maa-ala tuvastamine, servatuvastus, servade grupeerimine ning servaklastrite klassifitseerimine, millest viimane on samaväärne esemete tuvastamisega. Töö tulemusena pakutakse välja mitmeid uuenduslikke lahendusi, sealhulgas uus geomeetriline mudel maa-ala tuvastamiseks, kahe ääretuvastuse algoritmi koos kasutamine tõhusama ääretuvastuse saavutamiseks, ning hübriidne servade grupeerimise algoritm. Lisaks kirjeldatakse uut klassifitseerijat koos näidisesemete ja vastavate parameetritega. Töö on asjakohaselt illustreeritud selgitavate joonistega.

Võtmesõnad:

Ääretuvastus, grupeerimine, esemete tuvastamine, raalnägemine.

CERCS kood:

T111 Pilditehnika, T121 Signaalitöötlus.

Edge information based object detection and classification

Abstract:

This thesis presents work regarding the development 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. Numerous novel improvements are proposed, including 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. Also, a classification model is outlined along with example classifiers and used values. The work is illustrated graphically where applicable.

Keywords:

Edge detection, clustering, object recognition, computer vision.

CERCS code:

T111 Imaging, image processing, T121 Signal processing.

Contents

List of figures

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 [naob]. 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 [roba].

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 1.6GHz processor, making multitasking and complex procedures a challenge [naoa]. 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 [robb]. The NAO's main source of information is vision, provided by two cameras, each with a maximum resolution of 1280x720px. 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 [roba]. 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 [roba], both shown on 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

Figure 1: Sample frame from the robot's camera.

insufficient for a reliable model as demonstrated in [Bol15].

The aim of this thesis is to propose and implement the basis of a new edge information based vision module for use by University of Tartu's team Philosopher in the RoboCup SPL competition. The module will adhere to three main design principles:

- computation speed – information must be provided rapidly to enable the robot to make adequate decisions during the game;
- conservative use of resources – the NAO's single Intel Atom processor is shared by all its systems [naoa];
- universality – the module must work reliably regardless of fluctuations in lighting and noise.

To successfully implement the module, numerous topics will be covered: ground detection, edge detection, edge clustering and cluster classification. Each section will be analyzed from the perspective of the above main principles. As applicable, fitting approaches will be either chosen from existing solutions or new ones will be proposed.

Ground detection

In order to detect and classify different objects properly, identifying the area of the playing field currently 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 outlined in [SS09] and initial mean values proposed in [Bol15] for similar purposes, the green playing field area can be easily detected by setting a threshold value. This process, however, can leave many areas where the view may be obstructed by other robots excluded, as demonstrated on 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 Ψ_{raw}, the image is sized down to make all further operations computationally cheaper. The image is resized by a heuristically determined factor of 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, Ψ shall refer to the resized input frame.

$$\Psi_{\text{raw}} \xrightarrow{\frac{1}{9}} \Psi \tag{1}$$

Basic color thresholding is applied to Ψ based on values from [Bol15], yielding a binary array Ψ_B. All areas of set bits under a heuristically determined surface area are unset, reducing the amount of noise present in Ψ_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 on figure 5. The lowest corner points for the ground area are found:

$$\{L_l, L_r\} \in R \tag{2}$$

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:

$$\begin{aligned} \mid A_l^x - L_l^x \mid &< \epsilon_{\text{snap}} \\ \mid A_r^x - L_r^x \mid &< \epsilon_{\text{snap}} \end{aligned} \tag{3}$$

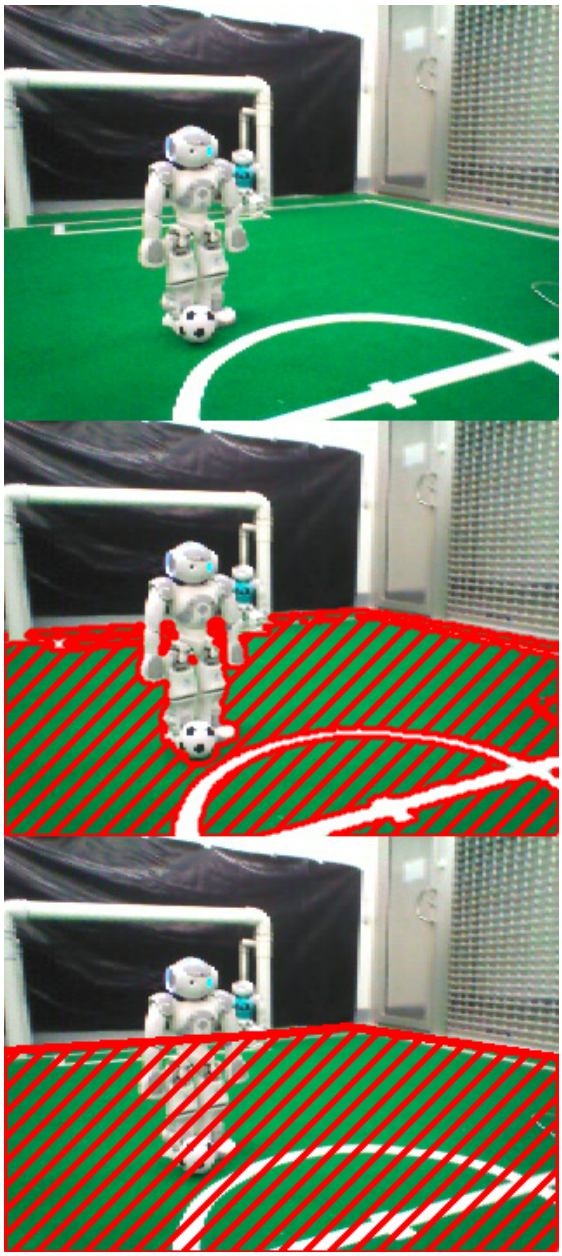

Figure 2: From top to bottom: video frame Ψ, naive color based ground detection, proposed method. Red marks the area considered "in the field".

where ϵ_{snap} is chosen heuristically. In practice

$$\epsilon_{\text{snap}} = 20\text{px} \tag{4}$$

Note that:

$$\{A_l, A_r\} \in R \tag{5}$$

Provided all work is conducted in the OpenCV standard Cartesian coordinate system [Lag11] demonstrated on figure 3, for each point in R, a weight ω is calculated by:

$$\omega^{(x,y)} = \begin{cases} \frac{\sqrt{y}}{\|\Psi(W-1,0)-\Psi(x,y)\|}, & \text{if } x < \frac{L_l^x+L_r^x}{2} \\ \frac{\sqrt{y}}{\|\Psi(0,0)-\Psi(x,y)\|}, & \text{if } x \geq \frac{L_l^x+L_r^x}{2} \end{cases} \quad (x,y) \in R \tag{6}$$

where W is the width of Ψ_B in pixels. For each half portion of the region between the bottom corners

$$\begin{aligned} S_l &= \max \omega^{(x,y)},\ x < \frac{L_l^x + L_r^x}{2} \\ S_r &= \max \omega^{(x,y)},\ x \geq \frac{L_l^x + L_r^x}{2} \end{aligned} \tag{7}$$

are then defined. While they overlap with previously found points in many generic scenarios, as can be seen on figure 6,

$$\exists \Psi_B : A_l \neq S_l \vee A_r \neq S_r \tag{8}$$

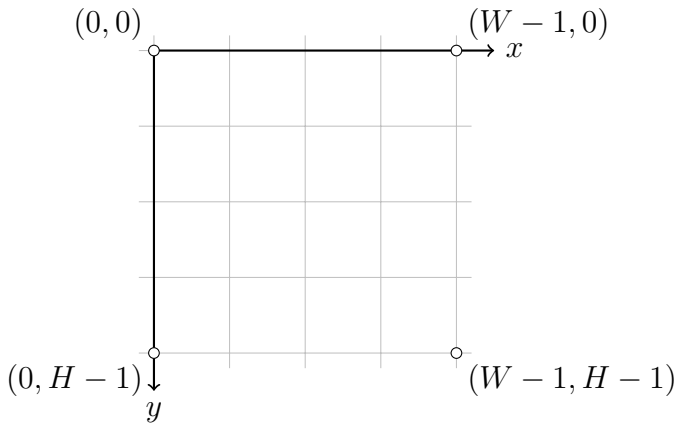

Figure 3: The OpenCV `cv::Mat` Cartesian coordinate system for an image Ψ with width W pixels and height H pixels.

Additionally, for each half portion of the same region, minima are selected by

$$\begin{aligned} M_l^{(x,y)} &= \min_y R,\ x < \frac{L_l^x + L_r^x}{2} \\ M_r^{(x,y)} &= \min_y R,\ x \geq \frac{L_l^x + L_r^x}{2} \end{aligned} \tag{9}$$

Finally, all points defined above are snapped to the closest edges of Ψ_B within a small threshold ϵ_{snap}, defined prior. This approach yields an 8-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 on figure 7.

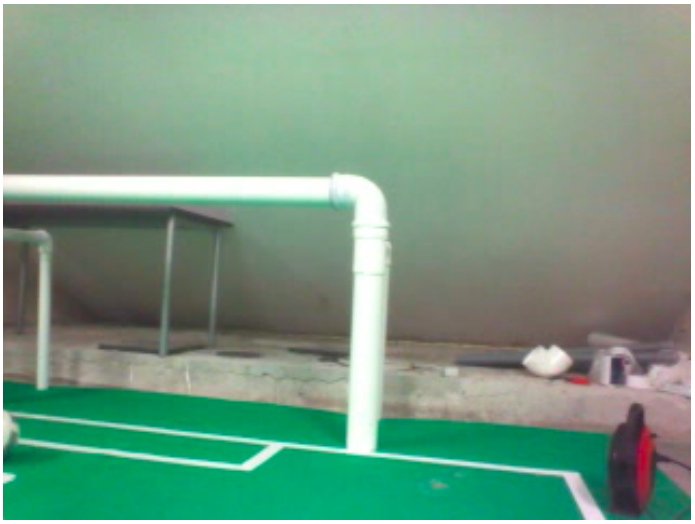

Figure 4: Video frame Ψ featuring obstructions and noise. Note a completely separated patch of green in the bottom right corner.

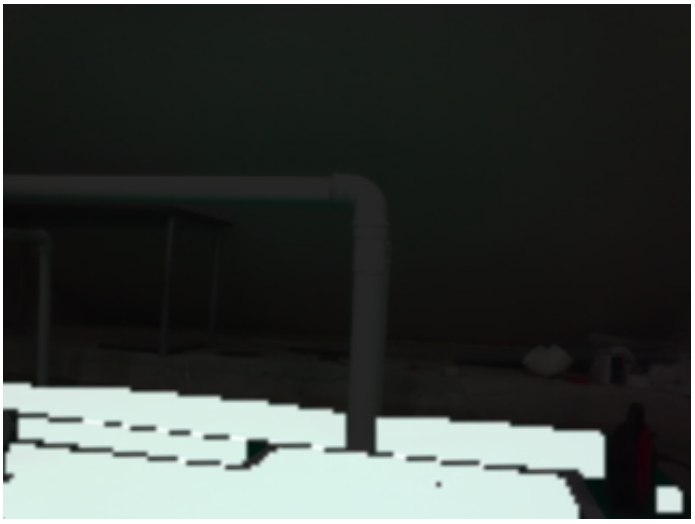

Figure 5: Binary thresholded frame Ψ_B with noise removed (rendered over the original frame Ψ with 85% opacity for reference).

The proposed method is a computationally cheap way ($O(N)$, where N is the size of Ψ) 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 Ψ:

$$F \subseteq \Psi \tag{10}$$

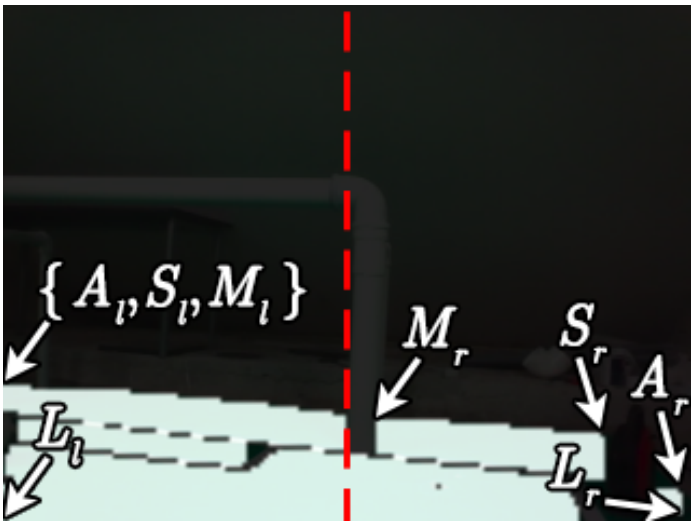

Figure 6: Annotated thresholded frame Ψ_B (ibid.) before snapping is applied. Red dashed line marks splitting defined by $\frac{L_l^x + L_r^x}{2}$. Note how $A_l = S_l = M_l$, but $A_r \neq S_r \neq M_r$.

Figure 7: The resulting polygon. Dashed red marks the area considered "in the field", i.e. dashed red marks F.

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.

Edge detection

Motivation

An edge $E \subseteq \Psi$ is a part of an image where significant variations in color intensity or brightness occur [Can86, OH10, Gom11]. Discontinuities in said properties generally correspond to changes in either depth, surface orientation, material properties or scene illumination [OH10], 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 [OH10]. 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 [ZT$^+$98]. 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, both are demonstrated on figure 8. Classical edge detection algorithms convolve an input image with a 2-dimensional operator O characteristic to that specific detector, yielding a grayscale response where edges are distinctively shown with either maxima or minima [Gom11]:

$$\Psi * O = \Psi_G \tag{11}$$

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 [SFM02]. 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_{D_1} and O_{D_2}, one for each diagonal direction [OH10]. Using two kernels yields two separate grayscale responses:

$$\begin{aligned} \Psi * O_X &= \Psi_{G_X} \\ \Psi * O_Y &= \Psi_{G_Y} \end{aligned} \tag{12}$$

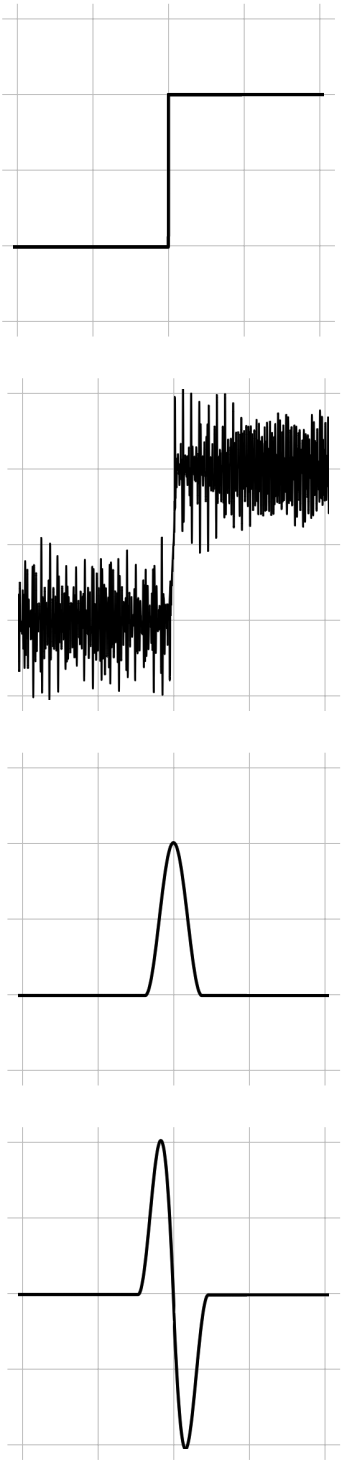

Figure 8: From top to bottom, not to scale: near-ideal sample step Φ, smoothed sample step affected by synthetic sinusoid noise Φ_N, first derivative Φ'_N of noisy sample, second derivative Φ''_N of noisy sample [ZT+98].

A very commonly used [SFM02] example of a gradient based approach is the Sobel operator, which uses

$$
\begin{aligned}
O_X &= \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \\
O_Y &= \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}
\end{aligned}
\tag{13}
$$

as its kernels [DH+73]. Given the above, the general gradient magnitude can be obtained by

$$
|\Psi_G| = \sqrt{\Psi_{G_X}^2 + \Psi_{G_Y}^2} \tag{14}
$$

commonly approximated instead by

$$
|\Psi_G| \approx |\Psi_{G_X}| + |\Psi_{G_Y}| \tag{15}
$$

as the latter is much faster to compute [Gom11]. Using separate kernels also makes edge directions easily computable from the responses, e.g. given responses from the aforementioned kernels [ZT+98]:

$$
\Psi_\theta = \tan^{-1}\left(\frac{\Psi_{G_Y}}{\Psi_{G_X}}\right) \tag{16}
$$

Other first derivative based methods compute gradient magnitude and edge direction in an analogous manner, with possible constants depending on kernel properties [Gom11].

Laplacian based methods rely on the Laplace operator Δ, which is a differential operator given by the divergence of a function's gradient in Euclidean space [VH01], given in two dimensions as [ZT+98]:

$$
\Delta\Psi^{(x,y)} = \frac{\delta^2\Psi}{\delta x^2} + \frac{\delta^2\Psi}{\delta y^2}\ , (x, y) \in \Psi \tag{17}
$$

For edge detection, approximate two-dimensional convolution kernels are used instead as the input space is discrete, e.g. [Gom11]:

$$
O_L = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix} \tag{18}
$$

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 Ψ. This makes computation cheaper, as for all common cases the size of O is considerably smaller than the size of Ψ [Gom11].

Canny

The edge detection approach proposed in [Can86], commonly referred to as "Canny edge detector"[Gom11], 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 [Can86]. 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 [OH10]. 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 [Can86]. Hysteresis in edge detection means tracking all edge candidates using two thresholds, a lower one and a higher one, ϵ_{lower} and ϵ_{higher} respectively.

$$\begin{aligned} \epsilon_{\text{lower}} &\in \mathbb{R} \\ \epsilon_{\text{higher}} &\in \mathbb{R} \end{aligned} \tag{19}$$

All points with brightness above ϵ_{lower} that can be connected to a point with brightness above ϵ_{higher} without any intermediate point having a value below ϵ_{lower} are set to full brightness, all others are suppressed [Gom11]. This means hysteresis can be used to map a grayscale input to a binary output:

$$\Pi_{\text{hyster.}} : \Psi_G \to \Psi_B \tag{20}$$

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:

$$\begin{aligned} \epsilon_{\text{lower}} &= \overline{\Psi_G} - \frac{\sigma(\Psi_G)}{2} \\ \epsilon_{\text{lower}} &= \overline{\Psi_G} + \frac{\sigma(\Psi_G)}{2} \end{aligned} \tag{21}$$

The approach yields accurate, consistent binary edges, demonstrated on figure 9. The result is improved further when histogram equalization has been previously applied to the grayscale input [RHP+06].

Canny's algorithm is the most commonly used edge detection algorithm due to its reliability, low complexity and availability [SFM02]. However, the algorithm is highly sensitive to fine detail, oftentimes more sensitive than required, and scenario specific parametrization is a prerequisite for good results [OH10]. The same problem applies to the current setting, as demonstrated on figure 9: 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 "Merged edge model".

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 [Bre01] consists of independently training a large of group 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. [GEW06] 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 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 [Bre01]. The algorithm is both fast [Bre01, DZ13] and, given a reasonably large training set, very accurate [GEW06].

[DZ13] 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 Ψ to an Euclidean space Z is used,

$$\Pi_{\text{simil.}} : \Psi \to Z \tag{22}$$

where comparing similarity can simply be done by comparing Euclidean distance. In order to avoid the issues outlined by [GEW06], 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

$$\exists\{P_1, P_2\} : P_1 \cap P_2 \neq \varnothing \tag{23}$$

The results are averaged across patches which can potentially lead to a general loss of accuracy. To counter the issue, [DZ13] runs the algorithm at multiple scales, labelling 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 [DZ15].

An edge detector based on random forests trained with the properties proposed by [DZ13] has characteristically soft edges as shown on figure 10. 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 [DZ13]. While this works well for general edge detection cases, in the given setting, crucial information may be lost in numerous scenarios as demonstrated on figure 10. This issue is addressed in a later section "Merged edge model".

Since the original implementation isn't available in any language used by the team Philosopher development team, the publicly available OpenCV implementation is used in it's stead. Even though the latter avoids the $O(2^N)$ baseline complexity implied by using double the input resolution for additional accuracy proposed in [DZ13] by instead running the algorithm only at original and half the resolution [ope], practical testing has shown that the implementation is still noticeably slower than the results presented in [DZ13, DZ15]. As no alternative comparable implementations are available currently, the problem is alleviated by running the algorithm on a previously scaled down input.

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 on figures 9 and 10. 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":

$$\Psi_{\text{Canny}} = \Psi \cap F \tag{24}$$

and the random forests edge detector is used to detect edges elsewhere:

$$\Psi_{\text{Rand.For.}} = \Psi \setminus F \tag{25}$$

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

$$\epsilon_{\text{overlap}} = 3\text{px} \tag{26}$$

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

$$\begin{aligned} \epsilon_{\text{higher}} &= \frac{1}{2} \\ \epsilon_{\text{lower}} &= \frac{1}{6} \end{aligned} \tag{27}$$

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 a following chapter.

The results from the two detectors, $\Psi_{B_{\text{Canny}}}$ and $\Psi_{B_{\text{Rand.For.}}}$, are then joined using bitwise or, annotated with $|$, resulting in a binary output, demonstrated on figure 11:

$$\Psi_B = \Psi_{B_{\text{Canny}}} \mid \Psi_{B_{\text{Rand.For.}}} \tag{28}$$

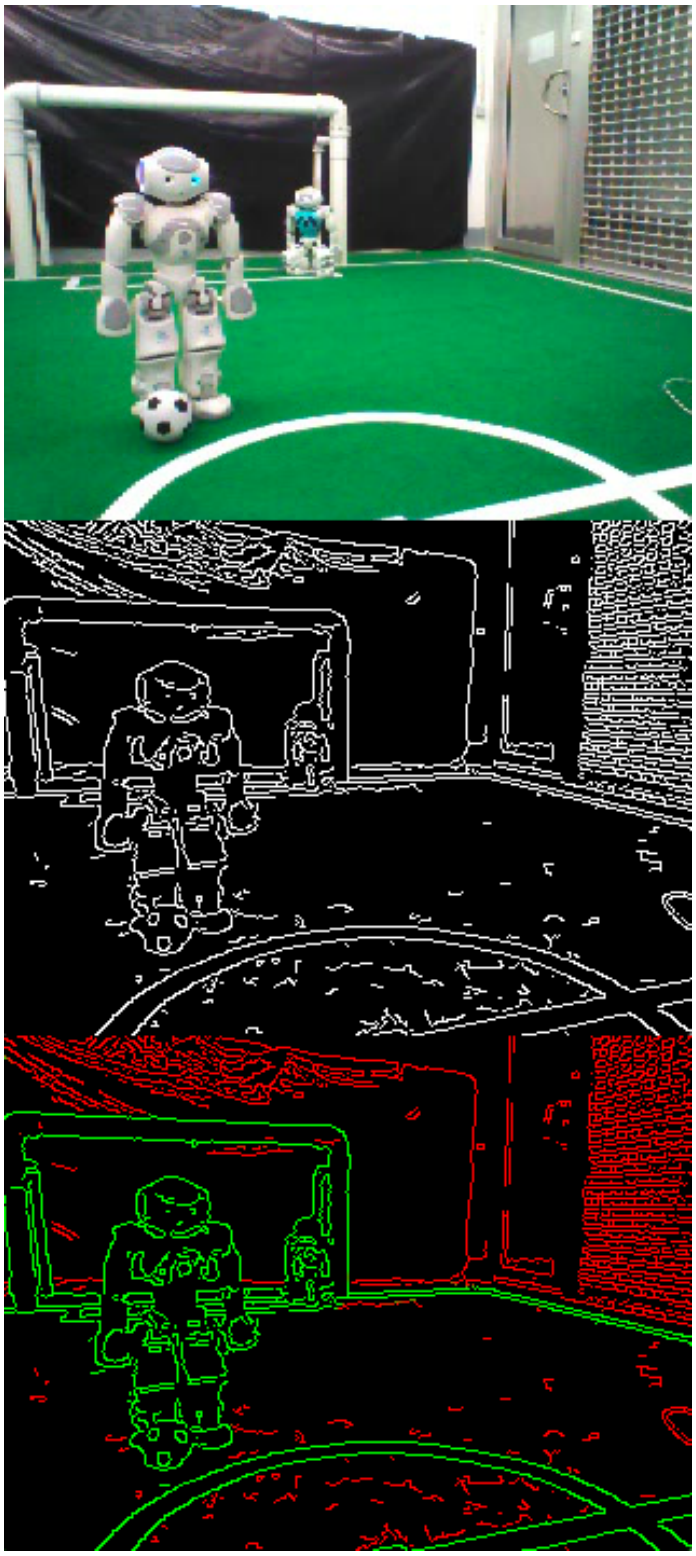

Figure 9: From top to bottom: video frame Ψ, binary frame Ψ_B from Canny's edge detector, manually annotated Ψ_B where green marks edges that offer valuable information and red marks all other edges.

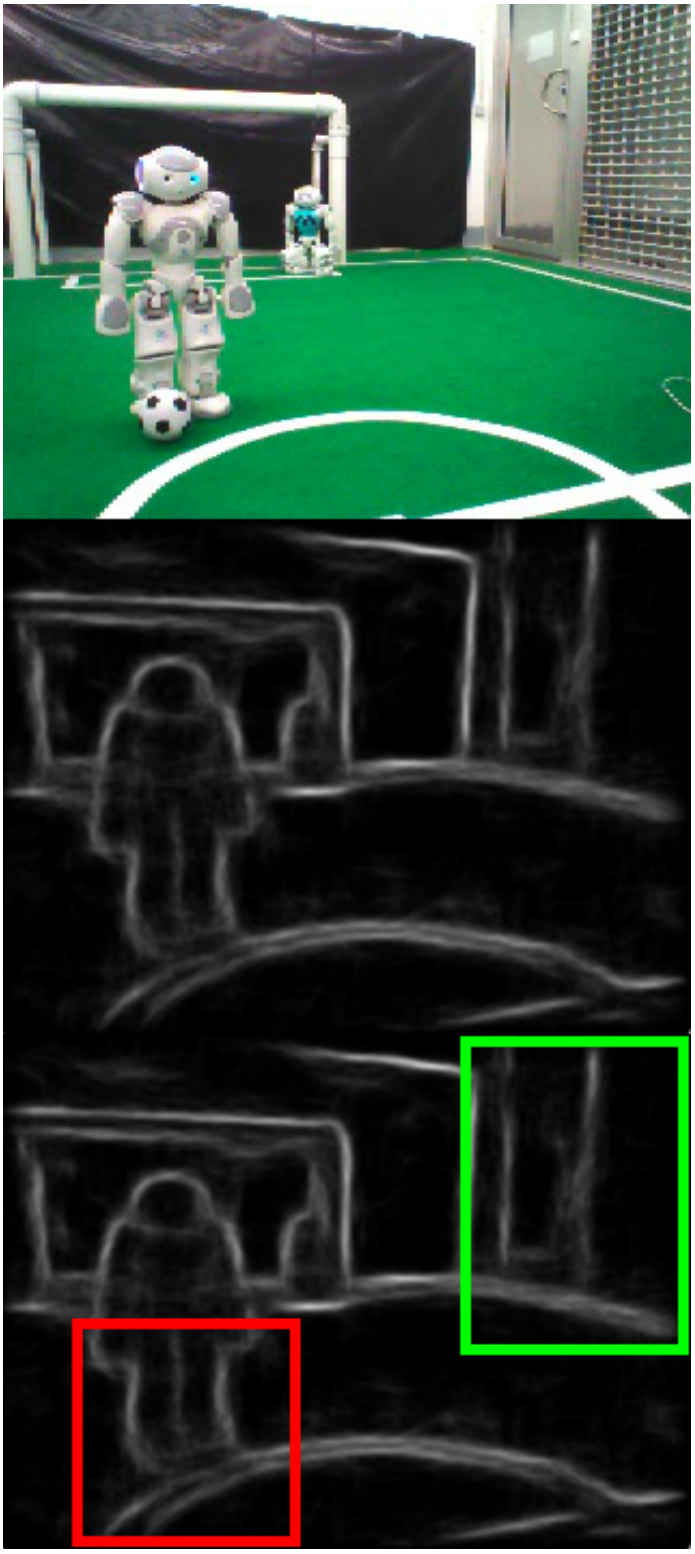

Figure 10: From top to bottom: video frame Ψ, grayscale frame Ψ_G from random forests edge detector, manually annotated Ψ_G where the red rectangle marks unwanted loss of information and the green rectangle marks beneficial loss of noise, both compared to Canny's edge detector.

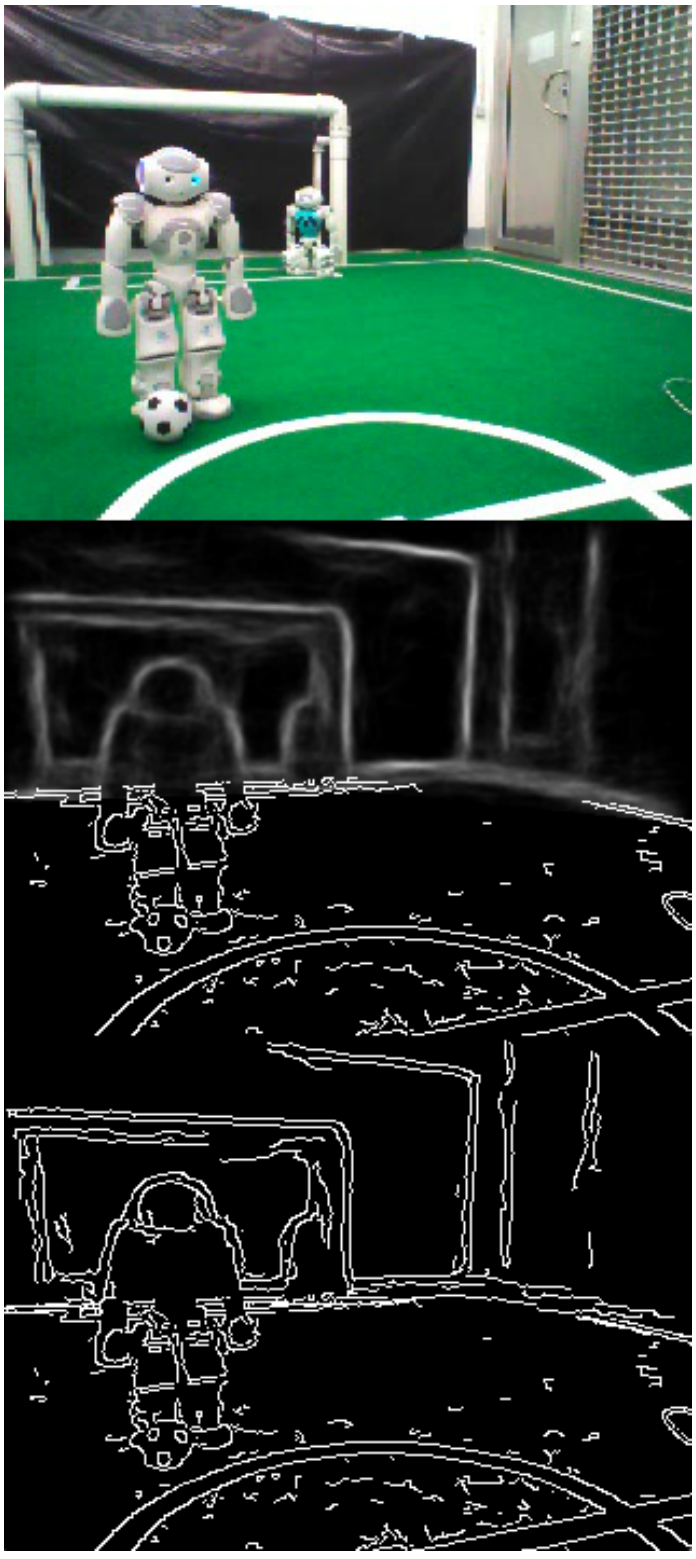

Figure 11: From top to bottom: video frame Ψ, grayscale frame Ψ_G resulting from combining the results of Canny's edge detector and the random forests edge detector without hysteresis, binary Ψ_B after hysteresis is applied.

Edge clustering

Each point $Q \in E$ in every edge $E \subseteq \Psi_G$ has a corresponding direction θ_Q that is equal to the gradient direction at that point, i.e.:

$$\theta_{\Psi_G(x,y)} = \Psi_\theta(x,y) \ , (x,y) \in \Psi_G \tag{29}$$

All directions are quantized into four categories analogous to the approach outlined in [Can86] and a label L is associated with every edge point using

$$L(Q) = \left\lceil \frac{4\theta_Q}{\pi} - \frac{1}{2} \right\rceil \bmod 4 + 1 \tag{30}$$

visualized on figure 12.

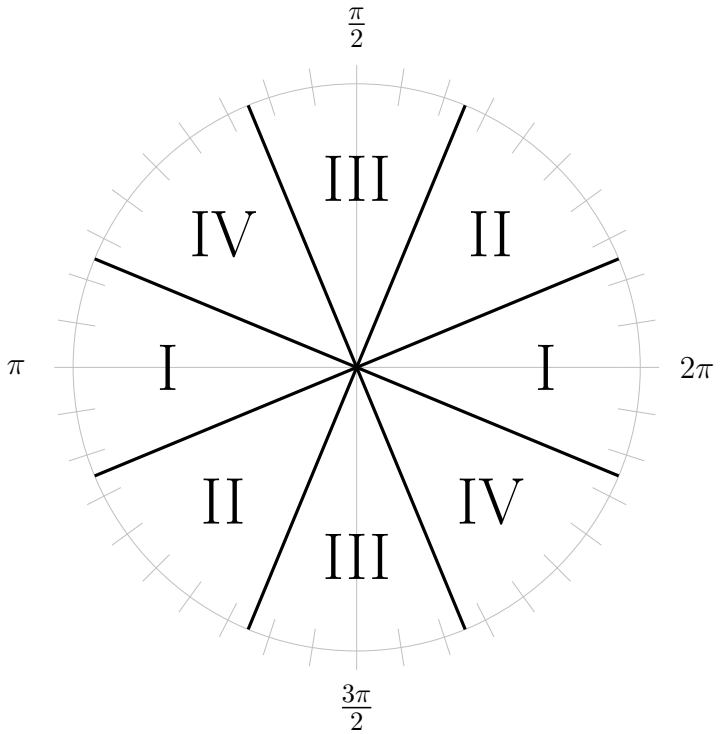

Figure 12: Quantized direction labels.

Similar to the approach proposed in [ZD14], edges are then grouped by joining all 8-connected points, except only edges with an identical label are joined, forming a cluster

$$C \subseteq E \tag{31}$$

demonstrated on figure 13. Using the relative direction difference proposed in [ZD14] 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:

$$\nexists Q : Q \in C_1 \wedge Q \in C_2 \tag{32}$$

Grouping is done recursively, proceeding along the horizontal axis of Ψ_G at first, as the memory addresses are sequential, and then vertically, as is the industry standard practice [Lag11]. Clusters with mass under a small heuristically determined threshold ϵ_{mass} are discarded as noise.

$$\epsilon_{\text{mass}} = 5\text{px} \tag{33}$$

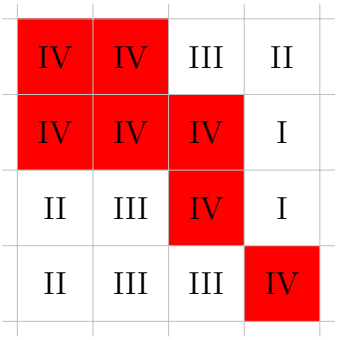

Figure 13: Sample patch of quantized directions, red marks possible cluster creation.

Classification of edge clusters

General overview

Multi-variable decision models have been shown to consistently outperform holistic, single descriptor classification models [SWP05]. 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 [Low04, BTVG06], some are too general and as such computationally too expensive, e.g. [RD06, RRKB11]. Using the proposed merged edge model with the clustering approach based on [Can86, ZD14], 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 [Low04, ZD14]: 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 on figure 14. Multiple weak classifiers working in unison make noise in any input variable less relevant [Bre01].

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.

$$\{Q_1, Q_2\} \in C \tag{34}$$

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.

$$\|Q_1 - Q_3\| = \|Q_2 - Q_3\| = \frac{\|Q_1 - Q_2\|}{2} \tag{35}$$

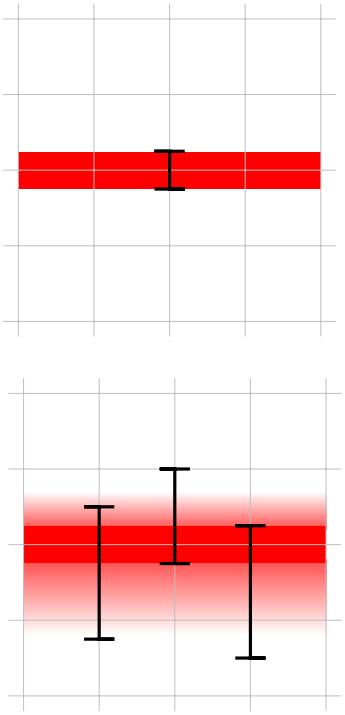

Figure 14: From top to bottom: a single strong classifier where a small deviation from the threshold values can mean a mismatch, a probabilistic collection of weak classifiers where noise in any input variable affects the final classification considerably less. Red marks classification matching, the vertical axis shows ranges for different variables, horizontal axis shows different classifiers.

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

$$\exists C : Q_3 \notin E \Rightarrow Q_3 \notin C \tag{36}$$

For all three points, brightness of Ψ 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 [OH10] is also employed.

Following, two sample classifications are described without probabilistic components, constructing a probabilistic model is considered out of scope for this thesis. All thresholds are determined heuristically given an input image Ψ, where Ψ is scaled down as outlined in "Ground detection".

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_θ, the average direction for the cluster must fall within

$$C_\theta \in (\frac{\pi - 0.4}{2}, \frac{\pi + 0.4}{2}) \tag{37}$$

C_{length}, the running length of the cluster, must be between

$$C_{\text{length}} \in (40\text{px}, 200\text{px}) \tag{38}$$

C_F shows how many of $\{Q_1, Q_2, Q_3\}$ for that specific cluster are considered "in the field":

$$C_F = |\{Q : Q \in \{C_{Q_1}, C_{Q_2}, C_{Q_3}\} \wedge Q \in F\}| \tag{39}$$

For initial goalpost candidates, this value must be

$$C_F \leq 1 \tag{40}$$

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:

$$\epsilon_{\text{connector}} = 20\text{px} \tag{41}$$

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 [Lag11].

Finally, all goalpost candidates that do not have a matching connector are removed from the list of matching candidates. Sample detections are shown on figures 15 and 16.

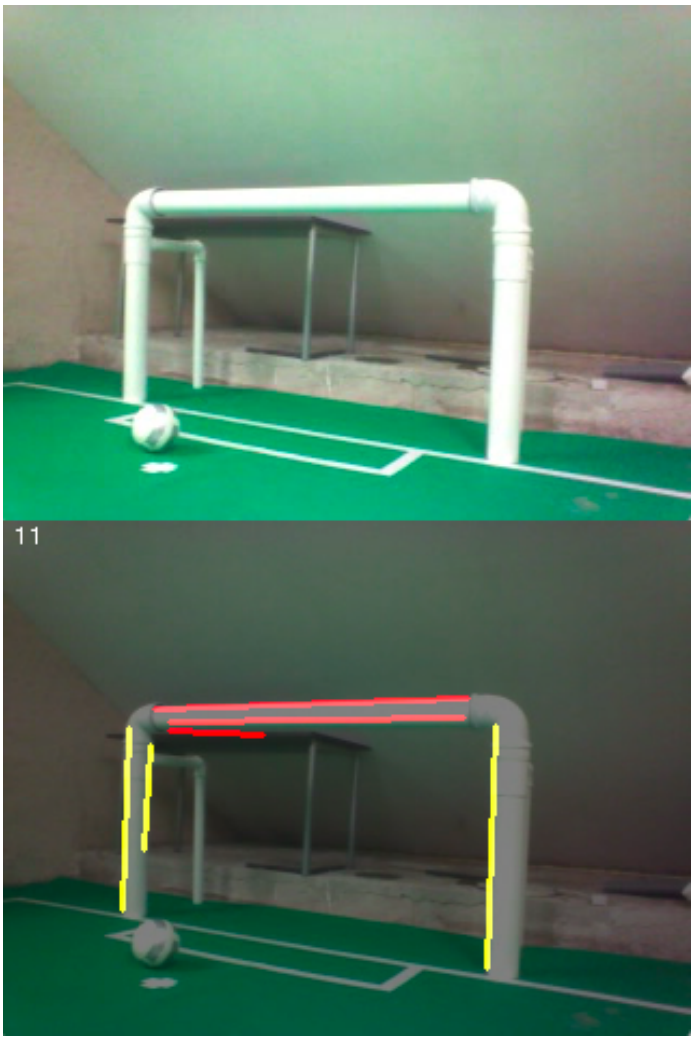

Figure 15: From top to bottom: original video frame Ψ, half opacity Ψ with annotations. Yellow marks detected goal posts, red marks detected top connectors.

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, the cluster's mass in pixels, must be under a given size

$$C_m < 20\text{px} \tag{42}$$

C_{sat}, saturation around the cluster's end points Q_1 and Q_2 in a bounding rectangle

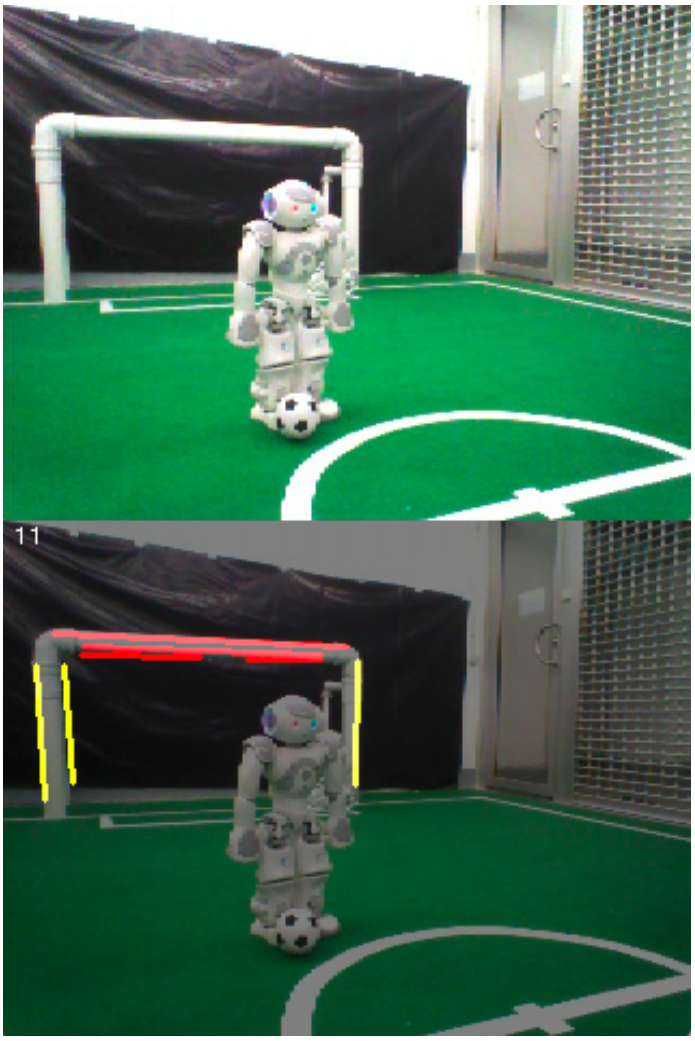

Figure 16: From top to bottom: original video frame Ψ, half opacity Ψ with annotations. Yellow marks detected goal posts, red marks detected top connectors.

with side $r_{C_{\text{sat}}}$ must not change more than a given value, i.e. assuming saturation values fall in $[0, 1]$:

$$\begin{aligned} C_{\text{sat}} &< \frac{1}{5} \\ r_{C_{\text{sat}}} &= 5\text{px} \end{aligned} \tag{43}$$

C_{bright}, the highest brightness value in a bounding rectangle with a side length $r_{C_{\text{bright}}}$ around both of the cluster's end points must be at least a given value, assuming all values fall in range $[0, 1]$:

$$\begin{aligned} C_{\text{bright}} &> \frac{4}{10} \\ r_{C_{\text{bright}}} &= 3\text{px} \end{aligned} \tag{44}$$

C_{dark}, the lowest brightness value with similar configuration must be below a given value

$$\begin{aligned} C_{\text{dark}} &< \frac{4}{10} \\ r_{C_{\text{dark}}} &= 3\text{px} \end{aligned} \tag{45}$$

C_F, described before, must be sufficiently high

$$C_F \geq 2 \tag{46}$$

These values reliably locate the ball on views where the ball is nearby, a sample detection is shown on figure 17. Locating the ball at all distances without a probabilistic classification model, described in section "Further work", is not feasible and is considered out of scope for this thesis.

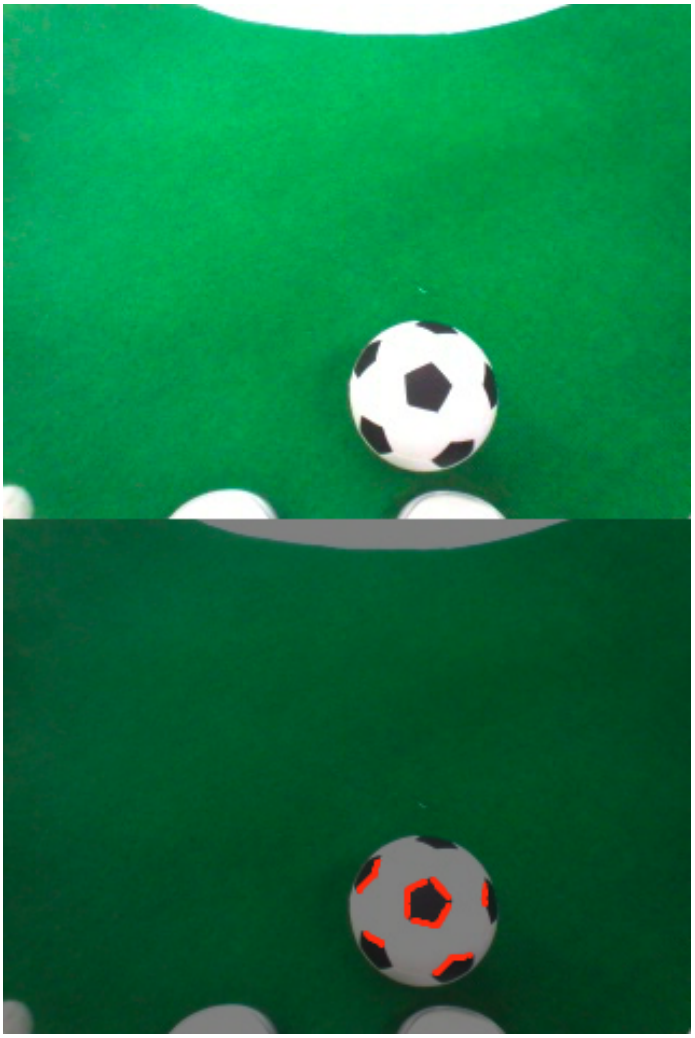

Figure 17: From top to bottom: original video frame Ψ, half opacity Ψ with annotations. Red marks clusters detected to be on the ball.

Conclusion and future work

Conclusion

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 [Bol15], improving the previous approach by removing any existing occlusions. A new merged edge model has been proposed that uses the algorithms outlined in [Can86] and [DZ13] in unison. As a prerequisite for object classification, an edge clustering method is proposed by combining the approaches from [ZD14] and [Can86]. 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 longer term outlooks for future research.

Future Work

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 [HCI+12], precision improvements outlined in [DZ15] and cluster merging outlined in [ZD14]. The binarization of the random forest edge detector can be improved by using the approaches outlined in [Can86]. The currently used random forest edge detection model is trained with parameters similar to those outlined in [Bre01] on the general BSDS500 dataset [AMFM11]. 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: the data is well structured and standardized, the problem space is well defined.

Bibliography

[AMFM11] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):898–916, 2011.

[Bol15] Anastasia Bolotnikova. Melioration of color calibration, goal detection and self-localization systems of nao humanoid robots, 2015.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer, 2006.

[Can86] John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.

[DH+73] Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

[DZ13] Piotr Dollár and Lawrence Zitnick. Structured forests for fast edge detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1841–1848, 2013.

[DZ15] Piotr Dollár and Lawrence Zitnick. Fast edge detection using structured forests. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(8):1558–1570, 2015.

[GEW06] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[Gom11] Abel Gomes. Visual computing and multimedia, 2011.

[HCI+12] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. Gradient response maps for real-time detection of textureless objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(5):876–888, 2012.

[Lag11] Robert Laganière. *OpenCV 2 Computer Vision Application Programming Cookbook*. Packt Publishing Ltd, 2011.

[Low04] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[naoa] Aldebaran NAO documentation. `http://doc.aldebaran.com/1-14/family/nao_h25/index_h25.html`. (Accessed: 12-05-2016).

[naob] Unveiling of NAO evolution: a stronger robot and a more comprehensive operating system. `https://www.ald.softbankrobotics.com/en/press/press-releases/unveiling%2Dof%2Dnao%2Devolution%2Da%2Dstronger%2Drobot%2Dand%2Da%2Dmore%2Dcomprehensive%2Doperating`. (Accessed: 19-05-2016).

[OH10] Mohammadreza Asghari Oskoei and Huosheng Hu. A survey on edge detection methods. *University of Essex, UK*, 2010.

[ope] OpenCV structured edge detection implementation source code. `https://github.com/Itseez/opencv_contrib/blob/master/modules/ximgproc/src/structured_edge_detection.cpp#L288`. (Accessed: 04-03-2016).

[RD06] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer, 2006.

[RHP+06] Mohamed Rizon, Yazid Haniza, Saad Puteh, Ali Yeon, Md Shakaff, Saad Abdul Rahman, Mamat Mohd Rozailan, Yaacob Sazali, Desa Hazri, and M Karthigayan. Object detection using geometric invariant moment. 2006.

[roba] RoboCup standard platform league. `https://www.informatik.uni-bremen.de/spl/bin/view/Website/WebHome`. (Accessed: 16-05-2016).

[robb] RoboCup standard platform league rules. `http://www.informatik.uni-bremen.de/spl/pub/Website/Downloads/Rules2016.pdf`. (Accessed: 16-05-2016).

[RRKB11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

[SFM02] Mohsen Sharifi, Mahmoud Fathy, and Maryam Tayefeh Mahmoudi. A classified and comparative study of edge detection algorithms. In *Information Technology: Coding and Computing, 2002. Proceedings. International Conference on*, pages 117–120. IEEE, 2002.

[SS09] Mohan Sridharan and Peter Stone. Color learning and illumination invariance on mobile robots: A survey. *Robotics and Autonomous Systems*, 57(6):629–644, 2009.

[SWP05] Thomas Serre, Lior Wolf, and Tomaso Poggio. Object recognition with features inspired by visual cortex. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 994–1000. IEEE, 2005.

[VH01] Ivan Matveevič Vinogradov and Michiel Hazewinkel. *Encyclopaedia of mathematics*. Kluwer Academic Publ, 2001.

[ZD14] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *Computer Vision–ECCV 2014*, pages 391–405. Springer, 2014.

[ZT+98] Djemel Ziou, Salvatore Tabbone, et al. Edge detection techniques-an overview. *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, 8:537–559, 1998.

All internet URLs were valid on 20.05.2016.

License

Non-exclusive license to reproduce thesis and make thesis public

I, Karl Tarvas (date of birth: 14.08.1992),

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

   1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

   "Edge information based object detection and classification", supervised by Assoc. Prof. Gholamreza Anbarjafari and Pejman Rasti,

2. am aware of the fact that the author retains these rights.

3. certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, 20.05.2016