Autonomous Navigation and Sign Detector Learning

Liam Ellis[1], Nicolas Pugeault[2], Kristoffer Öfjäll[1],
Johan Hedborg[1], Richard Bowden[2], Michael Felsberg[1]
{liam.ellis, kristoffer.ofjall, johan.hedborg, michael.felsberg}@liu.se
{N.Pugeault, R.Bowden}@surrey.ac.uk
[1] CVL, Linköping University, Linköping, Sweden [2] CVSSP, University of Surrey, Guildford, UK

Abstract

This paper presents an autonomous robotic system that incorporates novel Computer Vision, Machine Learning and Data Mining algorithms in order to learn to navigate and discover important visual entities. This is achieved within a Learning from Demonstration (LfD) framework, where policies are derived from example state-to-action mappings. For autonomous navigation, a mapping is learnt from holistic image features (GIST) onto control parameters using Random Forest regression. Additionally, visual entities (road signs e.g. STOP sign) that are strongly associated to autonomously discovered modes of action (e.g. stopping behaviour) are discovered through a novel Percept-Action Mining methodology. The resulting sign detector is learnt without any supervision (no image labeling or bounding box annotations are used). The complete system is demonstrated on a fully autonomous robotic platform, featuring a single camera mounted on a standard remote control car. The robot carries a PC laptop, that performs all the processing on board and in real-time.

1. INTRODUCTION

This article presents an autonomous robotic platform that learns to navigate on a track and discovers and detects road signs. The system consists of two modules handling steering control and sign recognition in real time. Both the driving behaviour and the sign detector are learnt from demonstration by a human tutor, rather than engineered. This is challenging as the important visual information is ill defined by the behaviour of the tutor and is therefore not a classical supervised learning task. Rather than basing control on scene reconstruction, Simultaneous Localization And Mapping (SLAM), or classical control techniques, the key idea of this work is to learn visual cues using control as a supervising signal.

It is generally assumed that the driver's actions are prompted by visual events such as the position of lane markers, other cars and traffic signs, and a common approach is to encode specific detectors for the visual events judged relevant. One issue with this approach is that all relevant events need to be known at the design stage, and the system will thereafter be unable to function in new unseen scenarios.

In this article we take a different approach to the problem, based on Learning from Demonstration (LfD), where the system is not given a-priori knowledge on the type of visual event it should expect or the driving rules it should follow, but rather learns them from observing a human tutor driving the system around the track, making use of state-of-the-art machine learning and data mining approaches. Thus, the system discovers what aspects of the visual percepts are predictive of the driver's actions for a given scenario.

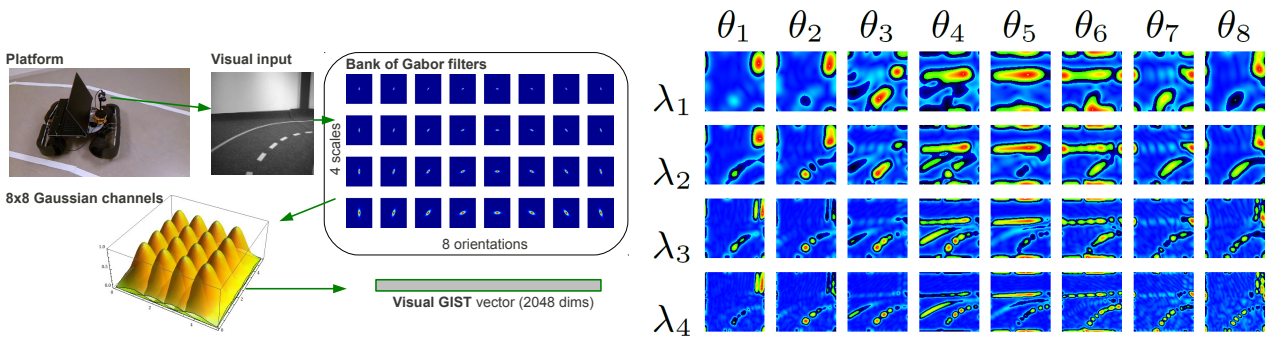
Related work: One of the earliest successful learning approaches to vision based autonomous navigation systems was ALVINN (Autonomous Land Vehicle In a Neural Network) [12]. Like the work presented in this article, ALVINN learns how to control a vehicle by observing a human driving. The learning is based on a single hidden layer back-propagation network. A more recent work from LeCun et al. focuses on learning vision based obstacle avoidance for off-road robots [9]. The learning algorithm in [9] requires very large collections of data and is based on a large 6-layer convolutional network. The system learns features that predict traversable areas.

Further literature, relevant to each component of the system is referenced in the corresponding section of this paper.

The rest of this paper is organised as follows: §2 gives a system overview, and details the training data collection strategy. Sections §3 and §4 detail the autonomous navigation and sign detection components, respectively. §5 contains experimental evaluations and §6 presents conclusions.

2. SYSTEM OVERVIEW

The system is based on a radio controlled car fitted with a laptop and a single grayscale camera. The laptop interfaces to the servo control on the car, so that controls generated



(a) Gist extraction (b) Gabor responses

Figure 1. Illustration of the visual Gist extraction process: (a) extraction of the Gist feature from the system's visual input; and (b) illustration of the Gabor responses at all scales $(\lambda_1, \ldots, \lambda_4)$ and orientations $(\theta_1, \ldots, \theta_8)$ for this image. In this figure, λ_1 is the coarsest scale and λ_4 the finest; θ_1 is tuned vertically and θ_5 horizontally.

by either the teacher (during training) or the software (at run time) are actuated. The autonomous navigation module generates control signals from visual input from the camera while the detector runs continuously to detect signs. The firing of the sign detector results in the issuing of a warning, and autonomy continues.

Training data collection: This work employs the approach of Learning from Demonstration (LfD), whereby policies are derived from example state-to-action mappings. Argall et al. present a comprehensive survey of robot LfD [2], that presents a categorisation of the multiple ways in which examples are gathered, including teleoperation and imitation. This work uses teleoperation to gather the examples, i.e. the robot is operated by a teacher while recording both the control signals and the sensor readings. The teacher controls the car with a standard remote control transmitter, by looking at a screen displaying images from a miniature wireless analogue camera attached to the robots camera. This ensures that the teacher is responding to the same percepts as are being observed by the system.

The obtained exemplar set is $E = \{(\mathbf{p}^i, \theta^i)\}_{i=1}^N$ where $P = \{\mathbf{p}^1 ... \mathbf{p}^N\}$ is the set of images and $\Theta = \{\theta^i ... \theta^N\}$ is the set of control vectors, with $\theta^n = [\theta^n_{\text{turn}}, \theta^n_{\text{speed}}] \in \Re^2$. From this exemplar set, control policies are derived that facilitate autonomous navigation, and additionally sign detectors are mined from the data with no additional supervision.

3. AUTONOMOUS NAVIGATION

The first module provides basic navigation, controlling the system's steering at a near-constant speed. The system does not have (or build) an internal model of the track, nor is it provided with prior knowledge of relevant features; instead it is expected to estimate the correct steering angle solely from its visual percept, that consists of a generic, holistic encoding of the entire visual field—also called Visual Gist. The vehicle is then expected to drive around the track indefinitely.

This problem of autonomous navigation has been approached in a number of different ways in the literature, which can be roughly divided between the classical control–based approaches [6, 20, 19] and the learning–based approaches [12, 13] (we refer to [10] for an in–depth review). The approach used in this article belongs to the second category, and draws from the LfD philosophy exposed in the introduction. Specifically, assuming consistency in lane markings and angles of curvature, we want to learn a mapping $f : \Phi \rightarrow \Theta_{\text{turn}}$ (where Φ denotes the visual features extracted from images P and $\Theta_{\text{turn}} = \left(\frac{\pi}{4}, \frac{\pi}{4}\right)$ the steering angles) that will keep the vehicle driving on the track indefinitely, for any track layout. Our aim is then to learn f from N example pairs $(\phi^i, \theta^i_{\text{turn}})_{i=1}^N$, sampled from a demonstration by a human driving around the track.

The chosen approach has the following characteristics: (i) the steering is estimated directly from the visual input, frame by frame and is therefore not affected by previous errors; (ii) the system does not have or build a model of the track, allowing for navigation on potentially infinite paths; (iii) the system's visual input encompasses the whole visual information, and it is the learning that specifies which aspects of the visual scene are relevant for steering control; and (iv) it runs real-time.

In the following we briefly describe the method used for learning this mapping: First we describe the visual feature, Φ, that the system uses to control steering; second we describe the regression algorithm. We refer to [15] for a more detailed technical discussion.

Visual features: The system's visual perception consists of a generic, holistic representation of the whole visual field, using so-called Visual Gist [11]. Visual Gist provides a compact and generic approach to image description, and has been used successfully for scene identification [16], image search [7], indoor vs. outdoor detection [17, 18], road type detection [8] and driver action prediction [14].

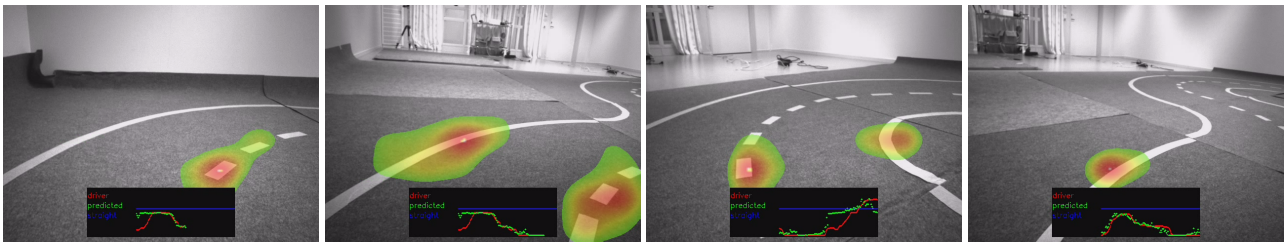


Figure 2. Illustration of the Random Forest activation when navigating around the P-shaped track. The red regions denotes visual areas that were mostly used by the regressor for predicting steering.

Visual Gist encodes the whole visual field into one feature vector, using multiscale filtering to encode the visual scene. There exists different versions of the Gist features. In this work, we compute the image Gist by filtering a downscaled (128×128 pixels) version of the image I with Gabor filters $G_{\lambda,\theta}(I)$ tuned at different scales (λ_i) and orientations (θ_i), and averaging the filters' responses over Gaussian channels $C_{i,j}$ regularly spaced over the image. We use 4 scales, 8 orientations and 8×8 channels, resulting in a feature vector of dimension 2048. This process is illustrated in Fig. 1(a), and Fig. 1(b) shows the Gabor responses to the same image, prior to the pooling in Gaussian channels. This shows how different filters capture different features of the visual scene.

Steering regression: In order to regress an approximation of the function f from the available examples, we make use of Random Forests (RF) [5]. Fundamentally, this comes to learning a population of randomized regression trees from random bootstrap samples of the original data. This has the effect of reducing overfitting, and considerably improves the performance compared to single, deep trees. Moreover, they are fast both at training and recall stages, and can be easily parallelized. Although RFs are commonly used for classification, we use them for regression of the steering angle. Effectively, regression trees learn by splitting recursively the input space at places that minimizes the variance of the samples on each side of the split. The process continues until a node contains few enough samples (≤ 5 samples in our case) or reaches a maximal depth (10 here)—forming a leaf node. Each tree leaf is then associated to an estimated steering angle, typically calculated as the mean of all samples that fell into this node. During retrieval, an input vector leads to a single leaf node per tree, and this tree's estimate is given by this node. The forest's estimate is the mean of all trees. In [15], it was argued that such an averaging leads to severe under-estimation of the regressed value, and it was suggested to use the medoid of all samples instead of the mean—this is coined RF-Medoid in the following. This intuition is confirmed in the results of section 5.

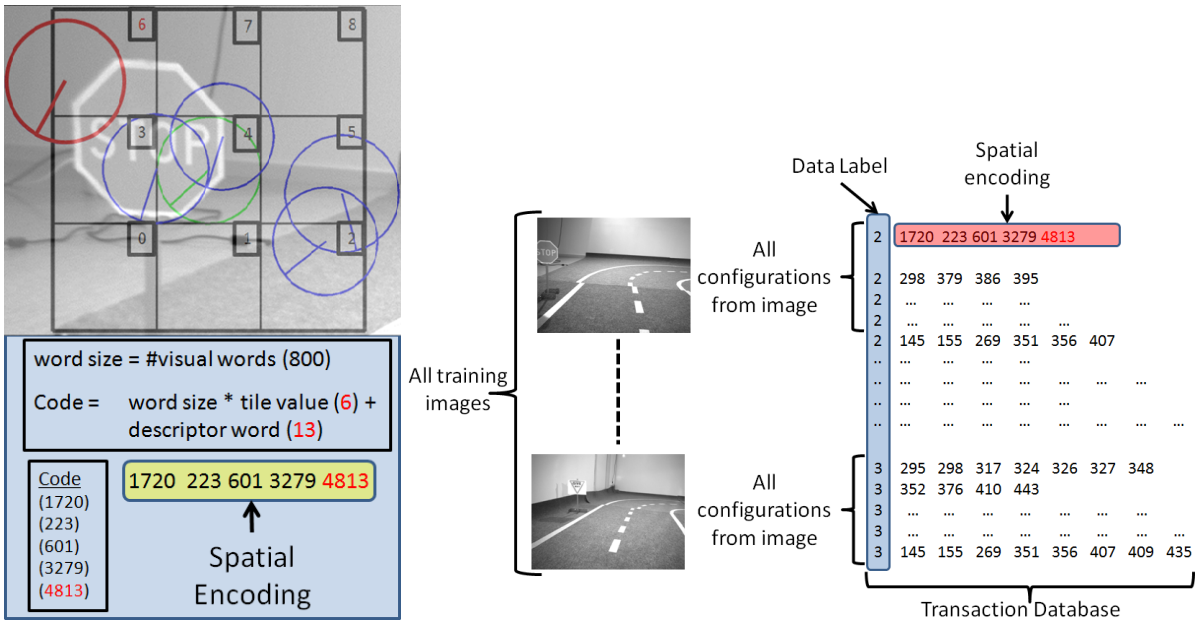
Activation maps: In order to illustrate the parts of the visual input that predicted the demonstrator's actions, we compute activation maps for the forests. An activation vector describes how frequently each input dimension was tested for reaching a tree leaf (normalized to one). The forest's activation is the mean of all trees. Such an activation vector can then be remapped onto the image, using the Gaussian channels of the Gist feature. Activation maps are illustrated in Fig. 2 and in the supplementary material. These show that the system attempts to follow the central dashed line of the track, and makes use of the two continuous outside lines when the road bends tightly. Note that the system was not provided with any prior knowledge of the lines configurations on the track and extracted them automatically based on their ability to predict the demonstrator's steering.

4. SIGN DETECTION

Beyond reactive behaviours (e.g. lane following), autonomous navigation and driver safety systems may be required to recognize visual entities, such as road signs, and to alter control policies or issue warnings. An interesting question is: how should descriptions of these entities, and their associated meanings, be imparted to the system? A standard approach entails constructing a detector (maybe using supervised learning), and hard-assigning a meaning to the sign (e.g. assign *Stopping Behaviour* to a STOP-sign detector). An alternative approach, investigated here, is for the system to discover visual entities that are strongly associated to modes of behaviour (e.g. *Stopping Behaviour* identifies the STOP-sign entity).

The proposed Percept-Action Mining approach learns detectors in a *weakly self-supervised* manner. Given the training set $E = \{P, \Theta\}$, unsupervised learning is applied to the control data, Θ, to identify **K** clusters, representing modes of behaviour, with labeling $L = \{l^i\}_{i=1}^N$. A weakly supervised learning algorithm then learns the sign detectors from the set $\{P, L\}$. For example, the clustering may identify a set of *Stopping Behaviour* exemplars. The images associated to this cluster are then mined to discover frequent and discriminative patterns. These patterns indicate the presence or absence of the visual entity that triggered the Stopping Behaviour, i.e. the STOP-sign.

The control data, is clustered - using k-means clustering



(a) Feature configuration encoding

(b) Labeled feature configuration database

Figure 3. Feature configuration mining for sign detection

- into $\mathbf{K} = 3$ clusters, relating to three behaviours, normal driving, stopping at a STOP sign and stopping and looking both ways at a GIVE WAY sign. In order to obtain invariance to displacement, scale and rotation, the control data is whitened prior to clustering. Even though these three behaviours are characterized by distinct profiles in the control data, and therefore form natural clusters, the resulting clustering is imperfect for the task, i.e. does not perfectly separate images with a certain sign from images without. This means that the discriminative pattern mining algorithm must handle noisy labels.

Given the weakly (and noisily) labeled training set $\{P, L\}$, a data mining algorithm is employed to discover frequent and discriminative patterns of local image features. This is achieved by building an image representation based on spatial configurations of visual words (quantised feature descriptors). This image representation includes the following processes: feature detection, feature description, codebook generation and spatial encoding. The SURF (Speeded Up Robust Feature) [3] image detector and descriptor is used to obtain local invariant image feature descriptors. SURF features are considered a good compromise between computational efficiency (required for real-time robotic applications) and robustness. A visual codebook is constructed by K-means clustering a subset of SURF features drawn from the training images $\{P\}$. An equal number (approx. ten thousand) of features are randomly sampled from each of the three subsets of $\{P\}$, identified by the labeling $\{L\}$. This ensures that the resulting codebook represents features drawn from images containing each sign, and background images. The resulting codebook contains 800 visual words (i.e. k=800 for K-means).

The final stage in constructing the image representation is to encode the spatial arrangement of visual words. The spatial encoding scheme is illustrated in figure 3(a). A 3 x 3 grid is centered on each SURF feature in an image, the grid is scaled to the size of the feature (the center feature illustrated in green), and the relative spatial arrangement of all local features that fall within the grid is encoded. The encoding scheme generates a single integer (code) that represents the existence of a particular visual word at a particular position within the 3x3 grid. For a codebook with 800 words, and a 3 x 3 grid, there are (800*8)+800=7200 possible integer codes.

Referring to fig. 3(a) (best viewed in colour), where the scale and orientation of detected SURF features are depicted by circles with orientation line, a 3 x 3 grid is scaled and located over the central (green circle) feature. The spatial arrangement of any other features that fall within the grid is encoded. For example the top left feature (red circle) is in grid tile 6 and has visual word 13. With 800 visual words, the spatial code for this feature is then 800 * 6 + 13 = 4813, as illustrated.

Spatial encodings are extracted from all training images, and a transaction database is constructed where each item in the database is the concatenation of the label $\{L\}$ of the image from which the spatial encoding is drawn, and the spatial encoding vector, as illustrated in figure 3(b)[1]. This

[1]The terminology *transaction* and *item* comes from the data mining literature, reflecting the subjects origins in market basket analysis applica-

database is then processed using the Apriori [1] data mining algorithm, in order to find frequent and discriminative feature configurations for each label. The Apriori algorithm exploits the anti-monotonicity of the support threshold constraint - that a subset of a frequent itemset must also be a frequent itemset - to efficiently mine association rules. This work uses an efficient existing implementation of the Apriori algorithm [4]. In this context, an association rule indicates a strong association between a particular spatial encoding of features, and a label.

A simple classifier is constructed from a set of association rules in the following manner: Spatial feature encodings are extracted from the input image, and are matched to the encodings that form the mined association rules. If sufficient (greater than a detection threshold) matches are made to a set of rules belonging to a class, then the image is classified as belonging to that class.

5. EXPERIMENTAL EVALUATION

The evaluation environment is a reconfigurable circuit made of tiles of carpet with sections of road markings. Different shaped circuits can be made by placing tiles in the desired configuration, see figs. 4(a) and4(b) for two example circuit configurations. To evaluate the quality of a trajectory of the robot around the circuit, the robot is tracked. Very accurate tracking trajectories were obtained by tracking a red marker attached to the robot. Figures 4(a), 4(b) and 4(c) show three different results of this red marker tracking (the red line). An homography is computed (by manually marking corresponding positions) from the position of the red marker to the ground plane as is shown in figures 4(a), 4(b) and 4(c) (the result being the green line). Finally a second homography is computed to project both the images and the trajectories into a plan view, see fig. 4(d). The following section details how these trajectories are evaluated, in order to compare the performance of various configurations of the regression algorithm with human performance.

5.1. Autonomous navigation performance

In order to evaluate the autonomous navigation performance, we considered two different circuit configurations: O-shape and P-shape. Performance was evaluated by estimating an *ideal* trajectory, following the dashed line in the middle of the road. This ideal trajectory is uniformly sampled at $N = 200$ points, $\hat{X} = (\hat{x}_1, \ldots, \hat{x}_j, \ldots, \hat{x}_N)$ and similarly for each lap the trajectory of the robot is sampled, $X_k = (x_{1,k}, \ldots, x_{i,k}, \ldots, x_{N,k})$. The distance between each robot trajectory point and the closest ideal trajectory point is calculated. The resulting error of a lap k around the track is given by

tions

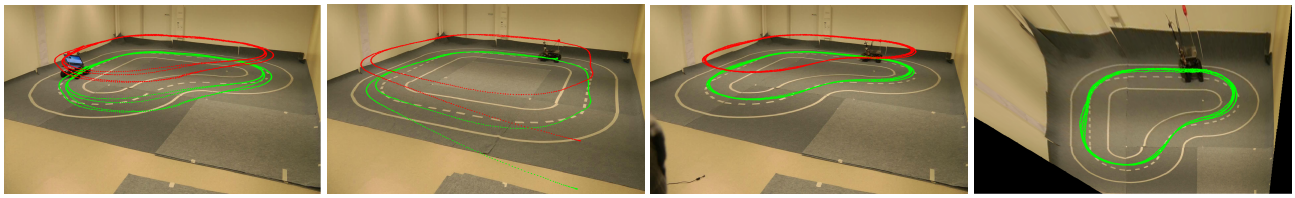
$$\epsilon_k = \frac{1}{N} \sum_i \min_j \|x_{i,k} - \hat{x}_j\| \quad (1)$$

We evaluated the mean and the variance of this error for $p = 4$ laps, and compare the performance of forests trained from 500, 1000 or 2000 training samples randomly picked from either O-shaped, P-shaped, or both circuit training data. The results are compiled in Table 1, where stars denote cases where the system failed to drive around the circuit. In the other cells small numbers (darker cells) denote smoother and better trajectories. The first row lists the error for an average of 4 laps of the demonstrator's trajectories. figs. 4(e) to 4(h) show all the trajectories of the human demonstrator when driving around the circuit. Note that the demonstrator's trajectories are neither consistent nor smooth, but succeed at driving the system around the circuit. In contrast the trajectories produced by the learnt mapping are considerably more consistent and smooth. This smoothness is a consequence of the stability of the regression model learnt. As the output of the model is the median of a large number of regression trees, the control signals are smoother than the training data.

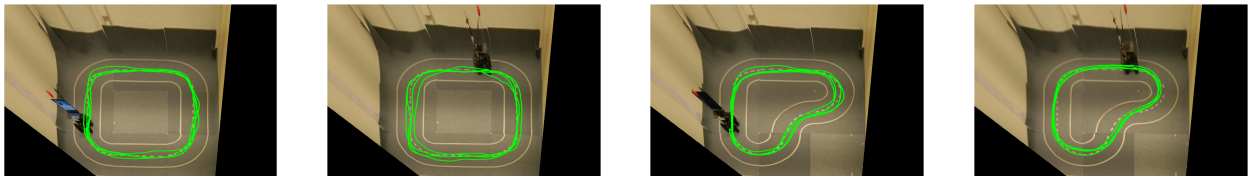
In all trials, using classical Random Forest regression based on mean lead to critical under-steering and all failed to navigate around the tracks (see figure 4(b) for the one partially successful sequence). In contrast, the use of RF-Medoid allowed the system to navigate successfully around the track—consistent with the results reported in [15]. In the following, we discuss the results obtained using RF-Medoid only.

Number of training samples: The first result that is apparent is that the performance degrades considerably when using only 500 training samples, causing total failure of the system to navigate around the track in most cases, and leading to large errors elsewhere. Performance increases significantly when considering 1000 training samples, leading the system to succeed at navigating around the track in most scenarios. Performance improves again, if less dramatically, when considering 2000 training samples.

Generalization performance: In order to evaluate generalization, we used two different tracks, O and P, and assessed performance on both when the system was trained on the same track, the other track, or both tracks jointly. As expected, the best performance is obtained when training on the same track, and the second-best when training on both tracks. We observe that training on the P-track allows to navigate the O-track successfully, whilst when training on the O track the system is unable to learn a model that can navigate the P track. This is due to the additional complexity of the tight curve that is absent in the O-curve, and shows that the learning can generalise to unseen track assuming that it has learnt from complex enough examples. This is probably due the the presence of concave curves in



(a) 1000 O-shape and P-shape samples (b) RF-mean rather than RF-median (c) 1000 P-shape training samples (d) Fig 4(c) plan view



(e) O-clockwise (f) O-anti-clockwise (g) P-clockwise (h) P-anti-clockwise

Figure 4. Evaluating steering angle regression by tracked robot trajectories

	O clock-wise			O anti-clock-wise			P clock-wise			P anti-clock-wise		
	500	1000	2000	500	1000	2000	500	1000	2000	500	1000	2000
human		**9.95**			**14.90**			**16.16**			**17.67**	
o	45.25	23.25	23.04	*	27.18	26.08	*	*	*	*	*	*
p	*	57.88	57.88	*	28.46	42.06	188.77	66.65	28.22	106.38	32.88	15.78
a	*	25.61	25.85	*	37.33	29.64	*	79.11	59.18	*	131.91	64.82

Table 1. Quality of the trajectories regressed by the system for different training sets (high is worse). We denote by $*$ cases where the system failed to drive around the track.

the PShape, that are absent in the OShape.

5.2. Mined sign detector performance

Figure 5 shows mined feature configurations correctly matched in test images. As can be seen the matched configurations vary in scale–the scale of the grid being dependent on the scale of the central feature, and also vary in complexity (number of spatial relationships encoded in each association rule). It can also be seen that not all features that make up a matched configuration necessarily lie on the sign (e.g. top right in fig. 5). This is an unavoidable result of the very weak supervision of the learning process, and is not always undesirable, as it encodes some contextual information that is present in the training data, for example, STOP signs are commonly found near sign posts.

A test set of 30 images (not seen during training) extracted from a sequence of the robot navigating a circuit with signs at the sides was manually labeled, with the number of images containing STOP and GIVE WAY signs being 10 each, with 10 background images (just containing the circuit). The number of feature configurations matched in all images for each detector was recorded, and an ROC curve generated, see fig. 5.2. For a false positive rate of 20% gives a true positive rate of 70% for the STOP sign, and 80% for the GIVE WAY sign, relating to configuration matching thresholds of 8 and 16 respectively. It is important to note that state-of-the-art sign detectors are trained on large datasets of hand labelled, tightly cropped on sign images, using expensive learning algorithms, whereas this algorithm requires no explicit sign label supervision and has learnt to detect signs even with a relatively small and noisy dataset.

As a test of run time performance, the robot was set to autonomously navigate around a circuit, with signs at the sides. To overcome occasional miss detections and false positives, a simple temporal filtering was applied, where the number of configuration matches is accumulated over a four frame window, and the detection threshold is applied to this sum. With this in place, the detectors achieve good performance, very rarely generating false positive or false negatives, as demonstrated in the supplementary material.

6. CONCLUSION

A learning based visually guided robotic system has been presented, for which both autonomous navigation and discovery of sign detectors are learnt using only tutor provided control as a supervising signal. Both tasks achieve good performance, with novel implementation.

The regression of GIST features onto control parameters using Random Forests is shown to achieve good generalisation - it can navigate circuits for which it has not been trained, and high accuracy and stability. It is also notewor-

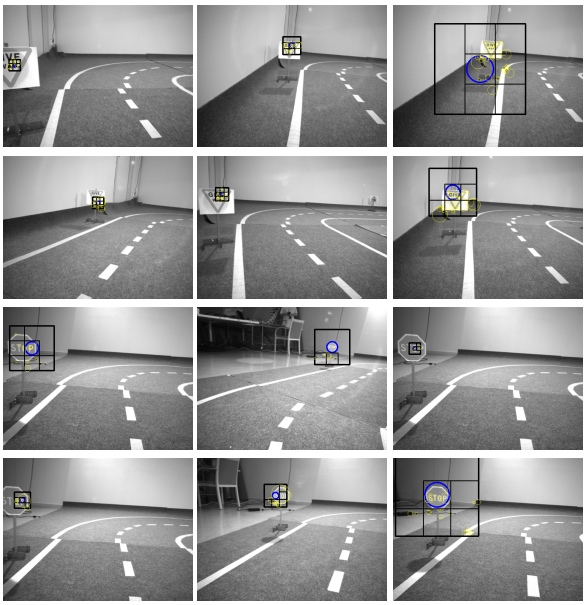


Figure 5. Mined feature configurations matching configurations in test data

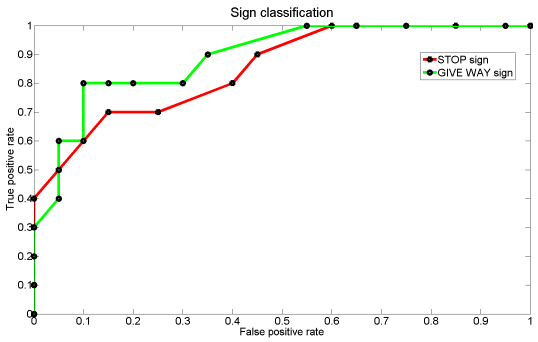


Figure 6. ROC curve for STOP and GIVE WAY signs

thy that the system achieves good generalisation, even with relatively little training data (1000 randomly selected samples for the Random Forest corresponds to approximately 1.5 laps of the PShape circuit), this is a significantly smaller training set size than is required by existing state-of-the-art learning based approaches (c.f. [9]).

The novel weakly self-supervised learning algorithm is shown to discover important visual entities and results in effective sign detectors. To the best of our knowledge, the approach of using control signals as weak supervision for detector learning is novel, and is a compelling example of the tight coupling between perception and action in visual recognition.

The system operates in real-time on a single laptop mounted on a standard RC car platform, by exploiting highly efficient algorithms for feature detection, large scale non-linear regression and classification.

Additionally, a novel and effective evaluation methodology has been established, that enables accurate measurements of navigation performance.

In future, the authors plan to work on including temporal reasoning into the framework, as well as higher level planning. Furthermore, the learning systems will be evaluated on an aerial robotic platform.

7. ACKNOWLEDGMENTS

This research has received funding from the Swedish Government for: ELLIIT (Strategic Area for ICT research), ETT (VR Swedish Research Council) and CUAS (Swedish Foundation for Strategic Research) and from the UK EPSRC: project EP/H023135/1.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robot. Auton. Syst.*, 57(5):469–483, 2009.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Vangool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[4] C. Borgelt. Efficient implementations of apriori and eclat, 2003.

[5] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] E. Dickmanns and V. Graefe. Dynamic monocular vision. *Machine Vision and Applications*, 1:223–240, 1988.

[7] M. Douze, H. Jégou, H. Sandhwalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *CIVR'09: Proceedings of the ACM International Conference on Image and Video Retrieval*, 2009.

[8] R. Kastner, F. Schneider, T. Michalke, J. Fritsch, and C. Goerick. Image–based classification of driving scenes by a hierarchical principal component classification (HPCC). In *IEEE Intelligent Vehicles Symposium*, pages 341–346, 2009.

[9] Y. Lecun, U. Muller, J. Ben, E. Cosatto, and B. Flepp. Off-Road Obstacle Avoidance through End-to-End Learning. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 739–746, Cambridge, MA, 2006. MIT Press.

[10] I. Markelic. *Teaching a Robot to Drive: A Skill-Learning Inspired Approach*. PhD thesis, University of Göttingen, 2010.

[11] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[12] D. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Proc. of NIPS*, 1989.

[13] D. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

[14] N. Pugeault and R. Bowden. Learning pre-attentive driving behaviour from holistic visual features. In *ECCV 2010, Part VI, LNCS 6316*, pages 154–167, 2010.

[15] N. Pugeault and R. Bowden. Driving me around the bend: Learning to drive from visual gist. In *1st IEEE Workshop on Challenges and Opportunities in Robotic Perception, in conjunction with ICCV'2011*, 2011.

[16] L. Renninger and J. Malik. When is scene identification just texture recognition? *Vision Research*, 44:2301–2311, 2004.

[17] C. Siagian and L. Itti. Rapid biologically-inspired scene classification using features shared with visual attention. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):300–312, 2007.

[18] C. Siagian and L. Itti. Biologically inspired mobile robot vision localization. *IEEE Transactions on Robotics*, 25(4):861–873, 2009.

[19] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L. E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney. Stanley: The robot that won the DARPA Grand Challenge. *Journal of Robotic Systems*, 23(9):661–692, 2006.

[20] M. Turk, D. Morgenthaler, K. Gremban, and M. Marra. VITS—a vision system for autonomous land vehicle navigation. *IEEE Trans. in Pattern Analysis and Machine Intelligence*, 10(3):342–361, 1988.