2017 IEEE International Conference on Robotics and Automation (ICRA)
Singapore, May 29 - June 3, 2017

Learning Robust Failure Response for Autonomous Vision Based Flight

Dhruv Mauria Saxena[1], Vince Kurtz[2], and Martial Hebert[1]

Abstract— **The ability of autonomous mobile robots to react to and recover from potential failures of on-board systems is an important area of ongoing robotics research. With increasing emphasis on robust systems and long-term autonomy, mobile robots must be able to respond safely and intelligently to dangerous situations. Recent developments in computer vision have made autonomous vision based navigation possible. However, vision systems are known to be imperfect and prone to failure due to variable lighting, terrain changes, and other environmental variables. We describe a system for learning simple failure recovery maneuvers based on experience. This involves both recognizing when the vision system is prone to failure, and associating failures with appropriate responses that will most likely help the robot recover. We implement this system on an autonomous quadrotor and demonstrate that behaviors learned with our system are effective in recovering from situational perception failure, thereby improving reliability in cluttered and uncertain environments.**

I. INTRODUCTION

Vision systems are known to be imperfect, which makes the vision system on any mobile robot prone to failures [1]. A number of different ideas have been explored in the robotics and computer vision literature that try to qualitatively assess the reliability of vision systems. Similarly, there is ongoing research that tries to predict failures in perception systems. However, we believe that it is equally important to make intelligent decisions once a failure has been predicted or recognized in order to mitigate any dangerous situations that might ensue. This is important to ensure long-term autonomy of mobile robots, and make them robust to the widespread situational changes that may occur in the environment the robot is operating in. An example of this from our system is shown in Fig. 1.

The task of associating failures with recovery maneuvers is challenging for three primary reasons. First, as robots become more robust due to advances in hardware and software, they will encounter failures less frequently. Second, associating failures with recovery maneuvers to some extent depends on domain knowledge related to the environment a robot operates in. This will cause different robots to possibly learn different recovery maneuvers. Finally, since mobile robots work on real-time data streams, they will almost certainly run into situations that could not possibly have been accounted for in any training set presented to the robot. It is therefore important to continue to learn from past experience for as long as possible. It is also

[1]The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. Email: dhruvsaxena@cmu.edu, hebert@ri.cmu.edu.
[2]Goshen College, Goshen, IN, USA. Email: vjkurtz@gmail.com.

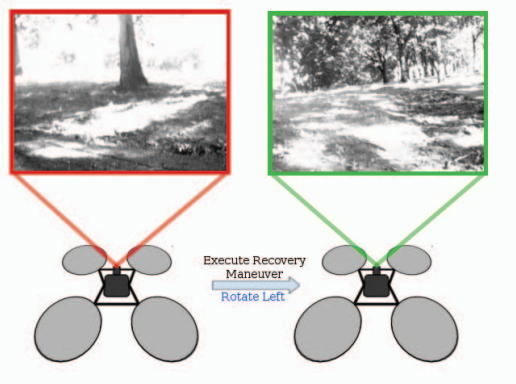


Fig. 1. After the quadrotor is alerted of a failure (possibly due to a combination of over-exposure and lack of features in the scene) on the left, our system tells it to execute the *rotate left* maneuver. At the end of this maneuver as shown on the right, the quadrotor has turned away from the source of illumination, and has enough information in the scene to continue its monocular flight. This is deemed a successful recovery from the perception failure.

difficult to manually label the different modes of failure. For mobile robot vision systems, over- or under-exposed images, motion blur, large inter-frame rotation, lack of texture, shadows etc. could all potentially cause failures. Fig. 2 shows example images from our actual flight tests that triggered perception failures, which were then resolved by one of the recovery maneuvers. The supervised learning task of classifying failures thus becomes intractable since it is almost impossible to label images in a training set with the cause of failure. A large number of factors could cause a vision system to fail. Rather than identifying the cause of failure, it then becomes more important to identify the recovery maneuver most likely to succeed in mitigating the failure.

An integral part of the framework presented in this paper is the ability of a robot to predict failures in their vision systems ahead of time. In the associated literature, this ability has been called *introspection* [2]. While a considerable amount of effort in the past has been dedicated to minimizing failures, predicting and identifying these failures during runtime has often been neglected. By extension, learning recovery maneuvers for these failures is a critical problem that we feel has not been given enough attention. The key contribution of this paper is the framework of data collection and associated training of classifiers that help us establish these relations between a failure and the best recovery maneuver. The set of recovery maneuvers is created by leveraging domain knowledge available to us. Creating this set from accumulated experience, or generating recovery

978-1-5090-4633-1/17/$31.00 ©2017 IEEE

(a) (b) (c) (d)

Fig. 2. Examples of images that failed during flight. Failures were resolved by (a) translating right, (b) translating left, (c) rotating right, and (d) rotating left. Intuitive causes of failure include direct glare from sunlight (c,d), strong shadows (b, d), open area (b,c), large obstacles preventing adequate parallax (a), and overexposure (a,c,b). Our algorithm used thousands of images like these to learn which trajectories are most likely to recover from perception failures.

maneuvers on-the-fly, is an open question that lies outside the scope of this work.

The setup that we are using for the problem and approach detailed in this paper is a quadrotor that uses monocular vision for autonomous flight through a cluttered environment. In our case, the intended environment is a dense forest, with roughly 2 trees per $4m \times 4m$ area. We have limited sensing in that there is no GPS available to us, nor can we globally localize ourselves in the environment since a precomputed map is also not available. In this respect, the widely studied field of SLAM in robotics is not directly utilized by our system.

II. RELATED WORK

The idea of introspection is central to the field of psychology [3]. The analogy in robotics is the model representation of a robot's current operational state[1]. This idea was first introduced by Morris *et al.* [4] who used it in an information-theoretic setting to improve a robot's decision making capability when it became uncertain of its operational state. Recently, the idea of introspection has been adopted for perception systems in terms of quantifying the predictive variance of classification and detection algorithms [5–7]. The idea there is to use algorithms cognizant of the fact that the assumption of independent and identically distributed (*iid*) data is usually not valid for real-world robotic systems. These algorithms can then be utilized in an active learning framework [7] to further improve predictor accuracy. This is in contrast to the one-shot learning [8] and zero-shot learning [9] paradigms that do not express predictor uncertainty and seek to maximize accuracy. All of the above work relies on looking at the output from a system to get a confidence estimate and in turn predict a probability of failure. The notion of introspection that we utilize in this work is best described by Daftry *et al.* [2] who obtain a confidence estimate by analyzing the input to the system. The key difference is that this approach makes it possible to quantify the reliability of input data which directly affects the quality of the prediction made.

[1]**Operational state** is a high-level representation of the configuration of various sub-processes in a robotic system. It should not be confused with an instance from the *state space* model of a robot. More details can be found in [4].

A closely related area of work concerns executing recovery maneuvers once a failure has been predicted. Previous work in this space has focused on the notion of *exception handling*, which aims to associate or learn recovery actions for specific failures [10, 11]. However, both of those works make strong assumptions about the cause of failure (which is known in both cases) and are thus able to associate them with precise recovery maneuvers. Our work makes no such assumptions, and instead aims to predict a recovery maneuver that is most likely to succeed. Verma *et al.* [12] utilize Bayesian state estimation and particle filters to identify failures, but make no attempt to recover from these failures. Perhaps the work most closely related to the work presented in this paper is that of Prasad *et al.* [13]. They use a reinforcement learning based framework to penalize state-action pairs that lead to SLAM failures. The key difference between their work and ours is that given that they operate in structured, indoor environments, they can leverage SLAM algorithms to obtain information about pose estimation and tracking errors. This information is not available to us directly, and so we cannot leverage SLAM algorithms to predict failures or potential recovery maneuvers.

III. VISION-BASED AUTONOMOUS FLIGHT

In this section we present details about the hardware and software used in our quadrotor for vision-based autonomous flight through a dense, cluttered forest. Details associated with learning failure responses are deferred until Section IV.

A. Setup

The primary hardware platform is a modified 3DR ArduCopter with an on-board quad-core ARM processor and a Microstrain 3DM-GX3-25 IMU. There are two monocular cameras on the quadrotor. A downward facing PlayStation Eye camera is used for real-time pose estimation. The image stream from a front-facing PointGrey Chameleon camera is relayed to the base station, where the perception and planning modules use it for monocular navigation. These modules use a semi-dense 3D reconstruction of the scene to select the optimal trajectory which is then sent back to and executed by the quadrotor. The quadrotor platform used for this work is shown in Fig. 3.

Fig. 3. The testbed: an autonomous quadrotor for flight through dense forests with a camera as the primary sensor [14].

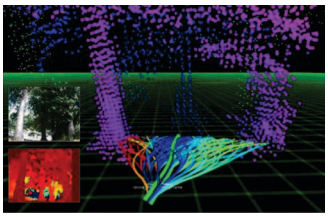


Fig. 4. Receding horizon control using a ground-truth depth image from a stereo camera. Trajectories in *red* have the greatest chance of collision, while the thick *light-green* trajectory is chosen as the best in this case.

B. Perception, Planning, and Control

The perception and planning modules run on the base station, while a pure-pursuit based PD controller is used for trajectory tracking on-board the quadrotor.

Mapping: We utilize a direct visual odometry based approach [15, 16] for semi-dense 3D scene reconstruction. This is then converted into an inverse-depth representation and propagated over multiple frames to obtain the final depth estimates for the pixels in the image [14].

Introspection: A deep spatio-temporal convolutional network, with architecture similar to AlexNet [17], is used to generate a good feature vector representation of the images from the front-facing camera's stream. These features are passed through a learned linear SVM which predicts a failure score between 0 (no failure) and 1 (failure) as output [2]. A failure of the perception system occurs when it is unable to reliably label the trajectories in our library as *collision-free* or *collision-prone*. The d-dimensional feature vectors $\mathbf{x}_i \in \mathbb{R}^d$ from the `fc7` layer of the spatial ConvNet for each image i are later used during training and inference of failure responses described in Section IV.

Planning and Control: We use a receding horizon planner that selects the current best trajectory out of a library of 78 optimally sampled motion primitives [18]. The best trajectory is selected such that a weighted sum of certain parameters is minimized. These include probability of collision along the trajectory, deviation from current heading, and deviation from desired/goal heading. These costs are calculated using the depth map obtained from the perception module. Once a trajectory has been selected, it is sent over to the quadrotor that uses a pure-pursuit PD controller to track the trajectories. A snapshot of the planner running on ground-truth stereo images is shown in Fig. 4.

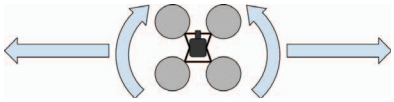


Fig. 5. We use four simple failure response maneuvers in this work: *translate right, translate left, rotate right,* and *rotate left*. These were selected by considering their ease of execution and resulting impact on the camera scene.

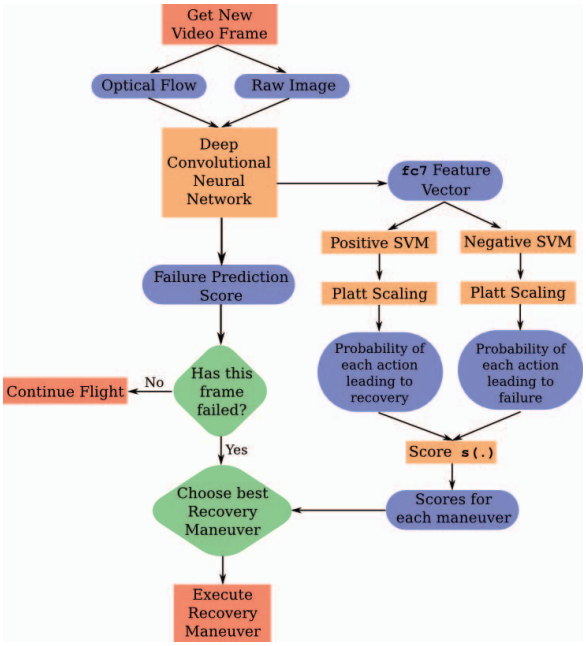


Fig. 6. Block diagram for learning failure responses. The key concept involved is learning the association between failures predicted by a deep introspection framework [2] and recovery maneuvers.

C. Failure Response

Simple maneuvers have the potential to resolve different types of perception failures, but which maneuver is best for a given failure is not always intuitive. The set of recovery maneuvers we use is shown in Fig. 5. Since these trajectories will later be used as target variables for two SVMs, we introduce some notation here. $y_i \in \mathcal{Y}$ represents a particular failure response associated with failure i, where $\mathcal{Y} = \{$*translate right, translate left, rotate right, rotate left*$\}$ represents the set of possible responses.

IV. LEARNING FAILURE RESPONSES

This section contains details about the entire pipeline involved in learning failure responses. This pipeline is shown schematically in Fig. 6.

A. Recovery Maneuvers

Maneuvers that will lead to recovery in a variety of failure cases must be chosen using domain knowledge before training the classifier. Maneuvers should be simple, interfere minimally with the robot's high-level task, not expose the robot to additional dangers while perception is unreliable, and provide a reasonable likelihood of ending the unfavorable conditions that led to perception failure. It

is obvious that the approach outlined in this paper requires at least two candidate maneuvers in order to make a decision about which one is better, but no other assumption is made pertaining to these maneuvers. The maneuvers themselves simply act as class labels for the predictors. For this work, the four recovery maneuvers considered were $\mathcal{Y} = \{$*translate right, translate left, rotate right, rotate left*$\}$ as shown in Fig. 5. During execution, these maneuvers are sandwiched between hover commands sent to the quadrotor. Both translate trajectories cause the quadrotor to translate in the respective directions for $5s$ with a linear velocity of $\pm 0.1m/s$ as desired. Similarly, the other two trajectories cause the quadrotor to yaw by $\pm 45^\circ$ with a constant angular velocity over $5s$.

B. Data Collection

Ideally, once a perception failure is predicted all candidate maneuvers would be executed and recovery status would be recorded for each maneuver. Practically however, we can only execute one of the maneuvers and record its recovery status. To collect training data, when a failure is predicted, only one of the candidate maneuvers is executed and the recovery status at the end of the maneuver is recorded. If the perception system recovers from the failure at any time during the maneuver, that maneuver is considered *recovered*. The completed data set then contains the failure, the maneuver, and whether that maneuver resulted in recovery.

Training data was collected by holding the quadrotor as we walked through a forest environment, executing one of the four maneuvers every time a failure was predicted. This *handflying* approach drastically reduced the time required to collect data, and was overall logistically easier to carry out than actual flight. The data was collected by handflying for over $20km$ in this way.

Even with $20km$ of handflying, only 825 failures were encountered: an insufficient number for training a robust predictor $y(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^d \rightarrow |\mathcal{Y}|$ that maps from a high-dimensional feature space $\mathbb{R}^d$ to the set of candidate maneuvers $\mathcal{Y}$. To solve this issue, multiple images from each maneuver were used in the training set. As soon as the system was alerted of a perception failure, all image frames were recorded until either the system recovered or the maneuver ended. With about 16 frames per second recorded, using every recorded frame would result in unhelpful redundant data [19]. To avoid repeats, redundant images were removed greedily, using the L_1 distance norm between `fc7` feature vectors $\mathbf{x}$ from the deep introspection framework as a similarity metric. Thus, a set of image feature vectors $\mathcal{X} = \{\mathbf{x}_i\} \forall\, i \in t = 1, \cdots, T$ obtained from a single maneuver is reduced to a set $\mathcal{X}' \subset \mathcal{X}$ such that

$$\mathcal{X}' = \{\mathbf{x} \in \mathcal{X} : |\mathbf{x}_i - \mathbf{x}_j| \geq \epsilon, \forall (i,j) \in |\mathcal{X}|, i \neq j\}, \quad (1)$$

where ϵ is a user-defined threshold. We chose ϵ such that $\frac{|\mathcal{X}'|}{|\mathcal{X}|} = 0.1$. $\mathbf{x}_i^k$ and $\mathbf{x}_j^k$ refer to the k^{th} elements of the d-dimensional feature vectors. With a slight abuse of notation, we will refer to the set $\mathcal{X}'$ obtained from one maneuver as $\mathcal{X}$ from now on.

TABLE I
IMAGES IN THE TRAINING SET

Maneuver	Recovered	Failed
Translate Right	745	631
Translate Left	738	530
Rotate Right	1280	863
Rotate Left	1234	1518
Total		**7539**

Distance in feature space was used because similarity between these deep features matters to the predictor, not pixel-wise similarity between the images themselves. L_1 distance in particular was used because it provides more accurate results in high dimensional space than traditional Euclidean distance [20].

C. Predictor Training

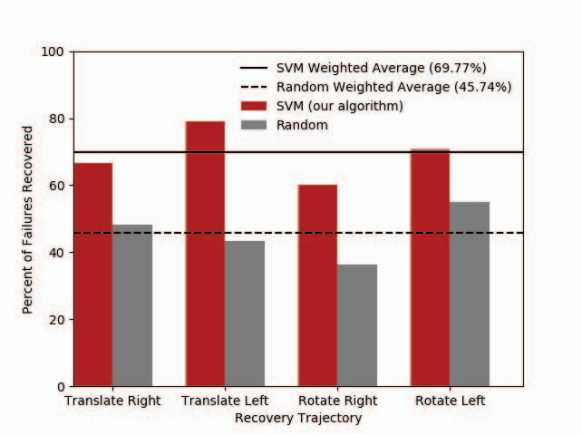
$\mathscr{X} = \{\mathcal{X}^1, \cdots, \mathcal{X}^N\}$ is the entire data set of images collected. The superscript i refers to the i^{th} maneuver that was executed. For the data that we collected, $N = 825$. This data set is split into two sets $\mathscr{X}^+$ and $\mathscr{X}^-$. $\mathscr{X}^+$ contains all images from trajectories that were successful in recovering from the perception failure, while $\mathscr{X}^-$ contains all images from trajectories that were unsuccessful in recovering from the perception failure. The exact numbers of images from each class $y \in \mathcal{Y}$ in each of these two sets is shown in Table I. Two SVMs are trained independently on these datasets $\mathscr{X}^+$ and $\mathscr{X}^-$, to predict the associated recovery maneuvers $y \in \mathcal{Y}$ which are used as the class labels.

We use the `fc7` feature vectors from the CNN independently for predicting potential failures, and selecting a recovery maneuver. The combination of CNN features and SVMs is a popular architecture for supervised learning tasks [21]. Using SVMs in combination with CNN features for these two independent tasks is a simple and more comprehensible model architecture (than an end-to-end neural network approach) that is able to outperform the existing state-of-the-art, as we discuss later in Section V.

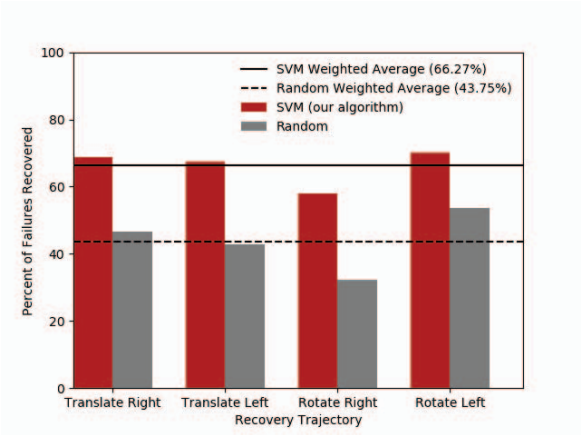
D. Predictor Inference

At test time, when a query image that triggered a perception failure is obtained, Platt scaling [22] is used to convert the arbitrary confidence scores from the two SVMs to two different probabilities. We represent $p^+(y|\mathbf{x})$ as the probability that the recovery maneuver leading to successful recovery from the perception failure is $y \in \mathcal{Y}$. Similarly, $p^-(y|\mathbf{x})$ represents the probability that the recovery maneuver leading to failure is y. Note that $\sum_y p^+(y|\mathbf{x}) = 1$ and $\sum_y p^-(y|\mathbf{x}) = 1$, but in general $p^+(y_i|\mathbf{x}) \neq (1 - p^-(y_i|\mathbf{x}))\ \forall\ i$.

We use the ratio of the two probabilities obtained from the two SVMs as the scoring function $s(\mathbf{x}) \in \mathbb{R}$ for each such query feature vector $\mathbf{x} \in \mathbb{R}^d$. The query image is then classified as belonging to the class with the greatest score.



(a)



(b)

Fig. 7. Results obtained from the experiments. The graphs show the percentage of failures that ended up recovery for each maneuver during (a) Handflight, and (b) Actual flight.

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} s(\mathbf{x}) = \arg\max_{y \in \mathcal{Y}} \frac{p^+(y|\mathbf{x})}{p^-(y|\mathbf{x})} \tag{2}$$

The predicted recovery maneuver $\hat{y}$ is then executed. $\hat{y}$ represents the recovery maneuver maximally likely to end in recovery and minimally likely to stay in failure.

V. EXPERIMENTS & RESULTS

A. Handflying

We first validated our approach by handflying the quadrotor for over $3km$. The predicted recovery maneuver was executed whenever a perception failure was encountered. Following this approach, 69% of failures ended in recovery, as compared to 45% when following a random maneuver as shown in Fig. 7.

B. Actual Flight

Finally, we carried out over $6km$ of tests in actual flight to test the robustness of this framework. Once a perception failure is predicted, the quadrotor stops following the previous trajectory it had received, and switches control over to the recovery maneuver to be executed. Upon completion of the maneuver, it resumes trajectory tracking.

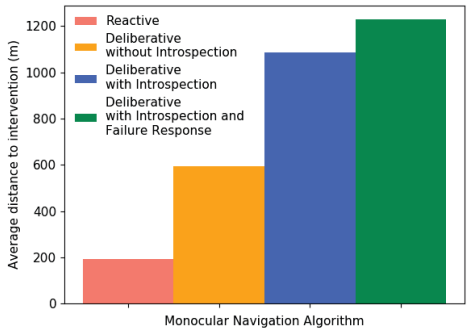


Fig. 8. Average flight distance without intervention in a dense, cluttered forest. The bars correspond to different algorithms from robotics literature - a purely reactive approach [23], a deliberative approach based on semi-dense monocular depth estimation [14], the same deliberative approach with introspection [2], and our framework which includes failure responses.

If however it did not recover, the maneuver is considered to have *failed*, and a new recovery maneuver is executed at this instant. Autonomous flight was intervened after 3 successive recovery maneuvers failed, or if the quadrotor flew dangerously close to an obstacle (tree, branch, bush etc.).

When the recovery maneuver was predicted by our novel framework, the quadrotor recovered from failure 66% of the time. In contrast, it only recovered 43% of the time if the maneuver was chosen randomly. Fig. 7 shows these results. In addition, with this framework in place, the quadrotor flew for over $1,200m$ on average through a dense, cluttered forest (roughly 2 trees per $4m \times 4m$ area) at $1.5\,m/s$ before requiring human intervention.

Fig. 8 is a graph comparing various approaches that have been used for monocular flight through comparably dense forests. The average distance flown by the quadrotor is over 6x greater than a previous reactive controller based approach [23], and over 2x greater than a naive deliberative approach without introspection [14] as shown in Fig. 8.

It has hard to compare our results with other existing research on autonomous outdoor flight [24–26]. The average distance flown without intervention, average flight speed, and density of obstacles in the environment are three metrics that are crucial for a fair comparison of attempts at autonomous outdoor flight. These works however either report only a subset of these metrics, or none of them.

VI. CONCLUSION

This work presents both a comprehensive argument for learning failure responses, and a framework which achieves that goal. During our experiments, it became evident that most perception failures were triggered due to improper illumination in the scene. The effects which caused failures ranged from over-exposure to strong-shadows. The use of illumination invariant images [27] for monocular navigation is thus an interesting research direction.

In this work, we used our domain knowledge to create the set of four candidate maneuvers. Our methodolgy during

training implicitly assumes that only one of these could recover from a failure, which is incorrect. This is evident from the fact that randomly selecting one out of four maneuvers results in recovery $> 25\%$ of the time as shown in Fig. 7. In future work, we hope to address the problem of selecting the *most likely to succeed* maneuver in a more intuitive way. Since only one maneuver can be executed per failure, and only its reward can be observed, this problem can be formulated more logically in a bandit setting [28], rather than the supervised learning setting considered in this paper. This would allow us to dynamically edit the set of recovery maneuvers (add/delete) based on the estimates of the mean rewards of maneuvers and the uncertainty around these estimates.

ACKNOWLEDGMENT

The authors would like to thank Arbaaz Khan and Sam Zeng for their help in collecting the training data and running experiments.

REFERENCES

[1] P. Zhang, J. Wang, A. Farhadi, M. Hebert, and D. Parikh, "Predicting failures of vision systems," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3566–3573.

[2] S. Daftry, S. Zeng, J. A. Bagnell, and M. Hebert, "Introspective perception: Learning to predict failures in vision systems," *arXiv preprint arXiv:1607.08665*, 2016.

[3] H. Kohut, "Introspection, empathy, and psychoanalysis: An examination of the relationship between mode of observation and theory." *Journal of the American Psychoanalytic Association*, 1959.

[4] A. C. Morris, *Robotic introspection for exploration and mapping of subterranean environments*. ProQuest, 2007.

[5] H. Grimmett, R. Triebel, R. Paul, and I. Posner, "Introspective classification for robot perception," *The International Journal of Robotics Research*, p. 0278364915587924, 2015.

[6] H. Grimmett, R. Paul, R. Triebel, and I. Posner, "Knowing when we don't know: Introspective classification for mission-critical decision making," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 4531–4538.

[7] R. Triebel, H. Grimmett, R. Paul, and I. Posner, "Driven learning for driving: How introspection improves semantic mapping," in *The 16th International Symposium of Robotics Research (ISRR)*. Springer, 2016, pp. 449–465.

[8] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.

[9] M. Palatucci, D. Pomerleau , G. Hinton, and T. Mitchell, "Zero-shot learning with semantic output codes," in *Neural Information Processing Systems (NIPS)*, December 2009.

[10] J. Searock, B. Browning, and M. Veloso, "Learning to prevent failure state for a dynamically balancing robot," in *AAAI*, 2005.

[11] R. R. Murphy and D. Hershberger, "Classifying and recovering from sensing failures in autonomous mobile robots," in *Proceedings of the National Conference on Artificial Intelligence*, 1996, pp. 922–929.

[12] V. Verma, G. Gordon, R. Simmons, and S. Thrun, "Real-time fault diagnosis [robot fault diagnosis]," *IEEE Robotics & Automation Magazine*, vol. 11, no. 2, pp. 56–66, 2004.

[13] V. Prasad, S. Singh, N. Pareekutty, B. Ravindran, and M. Krishna, "SLAM-Safe Planner: Preventing Monocular SLAM Failure using Reinforcement Learning," *ArXiv e-prints*, July 2016.

[14] S. Daftry, S. Zeng, A. Khan, D. Dey, N. Melik-Barkhudarov, J. A. Bagnell, and M. Hebert, "Robust monocular flight in cluttered outdoor environments," *arXiv preprint arXiv:1604.04779*, 2016.

[15] S. Daftry, D. Dey, H. Sandhawalia, S. Zeng, J. A. Bagnell, and M. Hebert, "Semi-dense visual odometry for monocular navigation in cluttered environment," in *IEEE International Conference on Robotics and Automation (ICRA) workshop*. IEEE, 2015.

[16] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European Conference on Computer Vision*. Springer, 2014, pp. 834–849.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[18] C. Green and A. Kelly , "Optimal sampling in the space of paths: Preliminary results," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-06-51, November 2006.

[19] P. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, pp. 78–87, Oct. 2012.

[20] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *International Conference on Database Theory*. Springer, 2001, pp. 420–434.

[21] Y. Tang, "Deep learning using support vector machines," *CoRR*, vol. abs/1306.0239, 2013. [Online]. Available: http://arxiv.org/abs/1306.0239

[22] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

[23] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning monocular reactive uav control in cluttered natural environments," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1765–1772.

[24] A. Giusti, J. Guzzi, D. C. Cirean, F. L. He, J. P. Rodrguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. D. Caro, D. Scaramuzza, and L. M. Gambardella, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 661–667, July 2016.

[25] A. J. Barry and R. Tedrake, "Pushbroom stereo for high-speed navigation in cluttered environments," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 3046–3052.

[26] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. PP, no. 99, pp. 1–1, 2017.

[27] W. Maddern, A. Stewart, C. McManus, B. Upcroft, W. Churchill, and P. Newman, "Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles," in *Proceedings of the Visual Place Recognition in Changing Environments Workshop, IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China*, vol. 2, 2014, p. 3.

[28] J. P. Mendoza, R. Simmons, and M. Veloso, "Online Learning of Robot Soccer Free Kick Plans using a Bandit Approach," in *International Conference on Automated Planning and Scheduling*, June 2016.