High-Fidelity Simulation for Evaluating Robotic Vision Performance

John Skinner[1], Sourav Garg[1], Niko Sünderhauf[1], Peter Corke[1], Ben Upcroft[1], Michael Milford[1]

***Abstract*—**
Robotic vision, unlike computer vision, typically involves processing a stream of images from a camera with time varying pose operating in an environment with time varying lighting conditions and moving objects. Repeating robotic vision experiments under identical conditions is often impossible, making it difficult to compare different algorithms. For machine learning applications a critical bottleneck is the limited amount of real world image data that can be captured and labelled for both training and testing purposes. In this paper we investigate the use of a photo-realistic simulation tool to address these challenges, in three specific domains: robust place recognition, visual SLAM and object recognition. For the first two problems we generate generate images from a complex 3D environment with systematically varying camera paths, camera viewpoints and lighting conditions. For the first time we are able to systematically characterize the performance of these algorithms as paths and lighting conditions change. In particular, we are able to systematically generate varying camera viewpoint datasets that would be difficult or impossible to generate in the real world. We also compare algorithm results for a camera in a real environment and a simulated camera in a simulation model of that real environment. Finally, for the object recognition domain, we generate labelled image data and characterise the viewpoint dependency of a current convolution neural network in performing object recognition. Together these results provide a multi-domain demonstration of the beneficial properties of using simulation to characterize and analyse a wide range of robotic vision algorithms.

I. INTRODUCTION

Robotic vision involves processing a stream of images from a camera attached to a robot moving through a physical environment. The pose of the robot's camera is controlled and is a function of previous images. The environment typically has a complex 3D structure with transient distractor objects of unknown type and motion, as well as time varying lighting conditions. An important consequence is that no robot vision experiment can ever be exactly repeated. Nor can the performance of different algorithms be easily compared, as they are in computer vision research, since they will be evaluated under different conditions: the robot's initial conditions may vary, as may the lighting, environmental conditions and camera path.

In contrast, progress in computer vision research is driven by datasets of images which are are static, not temporally related. There is no scope for an algorithm to request a slightly different view of the scene, as is possible in robotic vision. Computer vision research has recently made giant strides in performance through the use of deep learning, but this approach is fundamentally limited by the amount of representative real world image data that can be captured and labelled for training and testing purposes.

[1] Australian Centre for Robotic Vision, Queensland University of Technology, 2 George St, Brisbane, Australia

In this paper we propose the use of a state-of-the-art photo-realistic simulation tool, the Unreal Engine 4 by Epic Games, to address these challenges. This tool, developed for gaming, allows the creation of complex 3-dimensional worlds that are realistically rendered. The view from a camera at any arbitrary pose can be obtained, enabling us to mimic the motion of a robot through the environment, and the robot can have one or many cameras. We can also change the illumination conditions, adjusting the position of the sun, the cloud conditions, artificial light sources and atmospheric conditions such as fog. Because exact ground truth is known, camera poses and object positions estimated by robotic vision algorithms can be compared against the simulation state information.

The key contribution of this paper is studying the efficacy of this new tool for robotic vision, in particular evaluating algorithms in a systematic and repeatable manner, and its potential for generating large amounts of synthetic data for purposes such as training deep networks. We test our ideas in three common robotic use cases. The first is robust place recognition using the SeqSLAM[14] algorithm. We capture a reference path through the world at a particular time of day and then evaluate precision-recall performance (which we summarise as F1 score) for different paths through the world under a range of varying lighting conditions. The simulation allows us to change the path and the lighting independently – something that cannot be done when capturing image sequences from the real world. Secondly, for visual SLAM we evaluate the performance of the state-of-the-art OrbSLAM system in an environment where the ground truth is known (the simulation model and camera path) and the camera path and lighting conditions are varied. Thirdly, we look at object recognition. We use the simulator to exhaustively render camera observations of an object at every combination of pitch and yaw in small increments. We then use this data to evaluate the viewpoint independence of an exemplary state-of-the-art convolutional network.

The next section presents prior work in the area and introduces the tools that we use. Sections III - IV evaluate our approach for robust place recognition, visual SLAM and object recognition. Finally Section VI presents our conclusions and future work.

II. PRIOR RESEARCH

In this section we review the prior usage of simulation in robotics research and the two primary domains of place

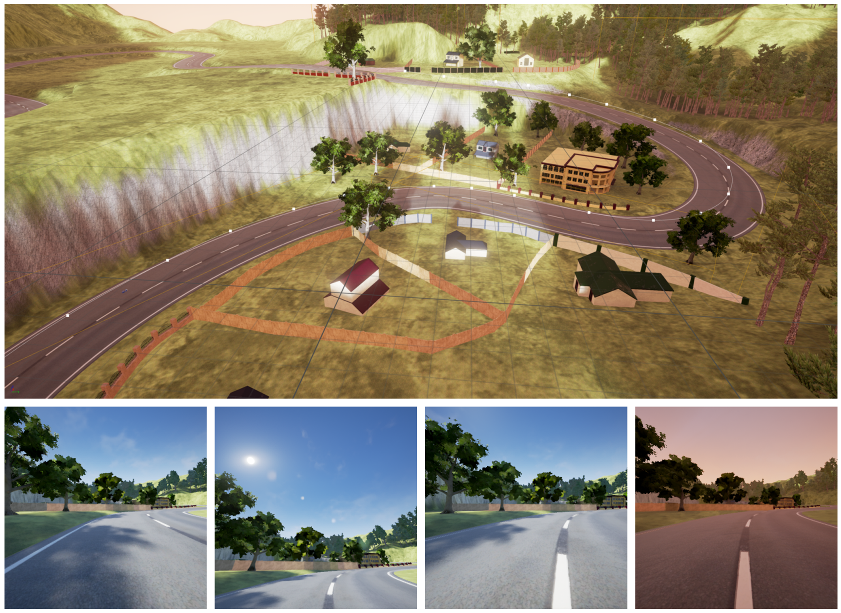

Fig. 1. The custom street scene used to capture the image datasets that were used for testing the place recognition algorithms. The line of white dots shows the baseline path followed by the camera when generating the datasets to test Sum of Absolute Differences and SeqSLAM place recognition performance. The SLAM experiments followed a slightly different path in order to ensure a complete loop of the environment.

recognition and visual SLAM we use to investigate the utility of simulation.

A. Simulation

The use of simulation in robotics is not in of itself a new idea. Popular robot simulators such as Gazebo [9] and Player/Stage [6] have existed for many years, and are commonly used to test robot motion and control. The use of game engines for robot simulation is also not new, with the USARsim project based on the Unreal Tournament 3 engine [2], and other projects using the Unity engine [10]. More modern, photo-realistic engines such as Unity 5 or Unreal Engine 4 (the successor to the Unreal Tournament 3 engine) are yet to be adopted widely by the research community, despite their significant advantages over previous generation technology.

Most robotic vision research has used image datasets captured from the real world. This is no doubt in part due to a natural skepticism of "‘simulation"’ results - the key requirement for simulation in computer vision is that the simulator is capable of photo-realistic rendering and lighting, which has historically been out of reach for older platforms, including the Unreal Tournament 3 engine or Gazebo. Unreal Engine 4 however has powerful tools for realistic materials and lighting [8], which make it possible to move a camera through a simulated environment and produce images similar to those obtained in the real world.

Unreal Engine 4 has a number of additional advantages that make it suitable as a simulator platform for computer vision. It is developed and maintained by Epic Games Inc, and uses physics and modelling tools from nVidia, which allow for complex and interactive dynamic environments. It also allows full access to its source code, allowing a simulation designer complete control over the simulation. It is also free for non-commercial uses, including research - an important property if it is to be widely adopted by the research community.

B. Place Recognition and Visual SLAM

A place is defined as a distinct 2D or 3D location in an environment. In robotic vision, visual places are described using the image features which can be broadly classified into local and global image descriptors. The most common and efficient approaches for recognizing a place revisited by a robot make use of Bag of Words approach with features, for example, SURF in FAB-MAP [4] and most recently ORB in ORB-SLAM [15] etc. Some extensions of such methods also include building vocabulary online in an incremental fashion or incorporating geometric constraints between words for better performance. These methodologies allow a wide baseline matching of places, but they are brittle towards vast changes in appearance of the environment. On the other hand, use of global image descriptors like BRIEF-GIST [22] or patch-normalized downsampled images as in SeqSLAM [14] allows matching across change in conditions, but lacks robustness towards viewpoint variations. There are some place recognition methods which have proven to work well with both condition and viewpoint variations as in [11], [13] and [17]. Some of the methods describe places in 3D using only monocular camera by employing Structure from Motion (SfM) techniques. This helps in sparse [15], semi-dense [5], [20] or dense [16] reconstruction of environment for visual SLAM and other similar applications.

The main challenges in place recognition lie in obtaining simultaneous robustness towards both change in conditions and viewpoint of a place. The environments with bland and texture-less images, motion blur, and effects introduced by camera properties like rolling shutter, granular noise etc. make it even more challenging to develop a high performance place recognition algorithm.

Visual SLAM systems typicallycomprise of a place recognition, visual odometry and mapping backend component. The overall challenges for such systems are related to consistently calculating camera motion between keyframes/frames, relocalizing camera position when tracking fails, handling dynamic changes in the environment, performing efficient loop closures to get rid of scale drift in the map, and efficient 3D map construction.

A robust place recognition or SLAM system needs to be evaluated against all the challenges mentioned above for a complete analysis of its performance. Such an in-depth analysis is often limited by a lack of variety in existing experimental datasets, or bias towards choosing datasets that work for the assumptions made by the particular algorithm. The existence of a high fidelity, fully controllable simulated environment provides a tool for overcoming these limitations and biases by performing thorough performance analyses.

There have been some past attempts towards generating simulated environments for evaluating robotic vision algorithms. Handa et. al. in [7] developed 3D models for living room and office room, and gathered camera trajectories for benchmarking RGB-D data based visual odometry, 3D reconstruction and SLAM algorithms. Peris et. al. in [18] created a simulated dataset for stereo systems for different illumination conditions. The work was inspired by the lack of adequate ground truth, especially the disparity maps for stereo vision. Similarly, Butler et. al. in [1] rendered images from an animated movie *Sintel* to create a dataset for evaluating optical flow methods. Ravi et. al. in [19] used a high fidelity marine simulator to test distributed cooperative 3D exploration algorithm for AUVs. Most of the work done in this regard has been focused on specific applications.

III. ANALYSIS OF PLACE RECOGNITION

To demonstrate the effectiveness of high-fidelity simulation as an analysis tool for robotic vision, we performed an in depth analysis of the viewpoint and time-of-day invariance of two different state-of-the-art place recognition algorithms, SeqSLAM [14] and OrbSLAM [15], and the viewpoint sensitivity of a state-of-the-art object recognition system.

A. Simulation Setup: Unreal Engine 4

The simulation tool used in this paper was the Unreal Engine 4, developed by Epic Games Inc. All of the test images used were generated from the same street scene, shown in Figure 1. The 3D models used were either sourced for free from TurboSquid (www.turbosquid.com), with significant manual clean up, or were produced manually. The landscape was produced using the basic version of World-Machine (http://www.world-machine.com). All lighting is as computed by the Unreal Engine, using standard sky and light assets provided with the engine.

To generate the image data, the camera was moved along a specified path to produce images at fixed spatial intervals. The simulator allows us to precisely repeat the same path and introduce calculated and precise variations upon it. In order to explore how the performance of place recognition changes with time of day and viewpoint change, passes along the path were generated for 5 different times of day in combination with increasing lateral offsets or camera tilts (Figure 2).

B. Place Recognition: Sum of Absolute Differences

The first set of tests evaluated a simple Sum-of-Absolute-Differences (SAD) based place recognition approach. While this is not a state-of-the-art technique in of itself, it has formed the foundation of major place recognition algorithms including RatSLAM [12] and SeqSLAM.

Before matching, each image is down-sampled to 64x64 pixels and reduced to greyscale, as per previous usage. The matcher used compares each image in a query dataset to each of those in a reference dataset, and considers the reference image with the lowest sum of absolute difference to be a match. The performance measure consists of the percentage of images for which the matched reference image is taken from a place close to the query image.

The matcher is tested using 130 different combinations of time of day and viewpoint changes, so that we can see how the performance falls off as both increase. The results are summarized in Figure 3.

Figure 3 shows that SAD-based matching seems to be relatively robust against small lateral offsets, with performance degrading beyond approximately $2m$of lateral shift. There is similar falloff as time of day approaches sunrise or sunset, and seems to be relatively symmetrical. The small number of samples makes this change seem relatively smooth, but it also seems plausible that it may instead change rapidly around dawn and sunset when the lighting change has the most effect. Resolving this is simply a matter of sampling the distribution further.

Interestingly, matching rate falls off similarly with angle change irrespective of the direction of change (compare the lower two plots in Figure 3). Matching performance seems to follow an exponential decay with angle. It may also be worth investigating the effects of multiple orientation offsets combined together to see how they compound. Were it the aim of this paper, it would be relatively simple to generate additional required sample passes to properly evaluate; for space reasons, we leave that to future work.

C. Place Recognition: SeqSLAM

The second place recognition algorithm we investigated was SeqSLAM, first described by Milford and Wyeth [14]. SeqSLAM is a place recognition algorithm that searches for loop closures by attempting to match sequences of similar images. It measures image similarity using sum of absolute differences as analysed in the previous section, and then searches for spatially coherent sequences of local best matches [14].

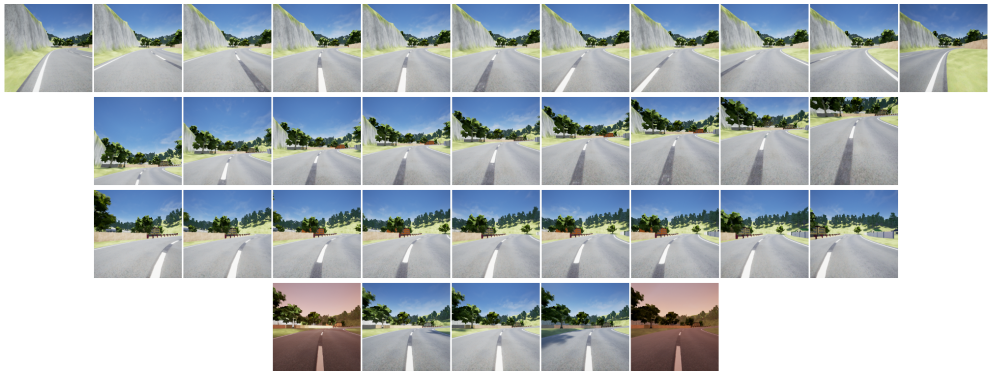

Fig. 2. A Sample of the variation found in the datasets used for testing. The first row shows lateral offset, from left to right, images are offset are left 3.5m, 2m, 1m, 0.4m, 0.2m, Then the baseline, then offset right 0.2m, 0.4m, 1m, 2m, and 3.5m. The second row shows vertical orientation change, left to right, images are angled are up 30°, 15°, 10°, 5°, Then the baseline, then angled down 5°, 10°, 15°, and 30°. The third row shows horizontal orientation change, from left to right, images are angled are left 30°, 15°, 10°, 5°, Then the baseline, then angled right5°, 10°, 15°, and 30°. Finally, the lowest row shows samples of time of day variation, from left to right, images are taken at dawn, in the morning, at noon, in the afternoon, and at sunset.

To perform the test, we generated 130 traverses across 5 times of day and 26 different viewpoint variations from the street scene used in the other tests (figure 2. We chose the baseline pass at noon as the reference dataset, and compared it to all 130 of the other datasets (including itself). For each dataset we calculated the maximum F1 score, with the results are summarized in Figure 4.

The first immediately obvious (and unpredicted) result is the anomalous low performance values for a left 0.2m offset in the morning and at sunset, and the low performance across all times of day at a 10°vertical orientation change. All of these changes are sudden and extreme, so the initial inclination is to sample around these points to see if there is a smooth or sudden decline. This analysis reveals that SeqSLAM performance degrades suddenly under these types of environmental changes.

The other feature of note in the performance characteristics is the way the F1 score plateaus for translational offsets of less than 1m and orientation changes of 5°. When the algorithm performs well, it seems to do so irrespective of the time of day. Strong condition invariance has been noted in previous work as a particular feature of SeqSLAM [14], which these results confirm. Note however that when the performance falls off due to lateral viewpoint change, it becomes less condition invariant, falling off further toward sunset and sunrise.

Using high-fidelity simulation has allowed us to explore the behaviour of SeqSLAM more comprehensively than ever before, and has revealed new details of its behaviour. This process could easily be repeated for an even wider range of environment permutations, identifying algorithm weakpoints and enabling future research improvements.

D. Visual SLAM: ORB-SLAM

The SLAM algorithm tested was ORB-SLAM [15]. ORB-SLAM is a feature-based monocular SLAM system, that performs feature-based visual feature tracking, place recognition, mapping and loop closure using ORB features.

To test OrbSLAM, we again generated a variety of image datasets from the same street scene, with 5 different times of day, a baseline pass and 20 different viewpoint variations. Due to the mapping element of OrbSLAM, all datasets were made to start and end in the same place. To test the performance, each of the datasets was appended to the noon baseline dataset (acting as a reference pass), and camera trajectories were generated. These trajectories were then compared with the ground truth to calculate the Absolute Trajectory Error after aligning their scale. Results are summarised in Figure 5.

The observed performance of ORB-SLAM is patchy, with inconsistent errors with no clear correlation to viewpoint or condition change. Poor performance is obtained on some of the baseline datasets with no viewpoint change, such as the morning pass. On the other hand, the performance of the algorithm doesn't seem to depend on viewpoint or time of day at all, but on other factors not controlled in our data.

E. SeqSLAM parameter tuning

The detailed analysis we have performed can be used to inform subsequent actions, including parameter tuning. To demonstrate this, we again tested the performance of Sum of Absolute Differences on the noon datasets across the range of lateral offsets shown in Figure 2 with a variety of different matching offset window parameters. SeqSLAM uses Sum of Absolute Differences combined with an offset window parameter that shifts the image pixels to achieve a better matching image. The results of this test can be seen in Figure 6.

This data suggests that for distances up to 3.5m offset from the reference location, it is best to use the smaller 4 pixel window range. The use of an offset window shows clear improvement in performance, but larger windows introduce false positive matches.

Benefits of Simulation: The performance curves generated using simulated datasets for SeqSLAM show a peculiar behaviour for certain parameter values. This enables us to focus analysis of the algorithm on these unexpected

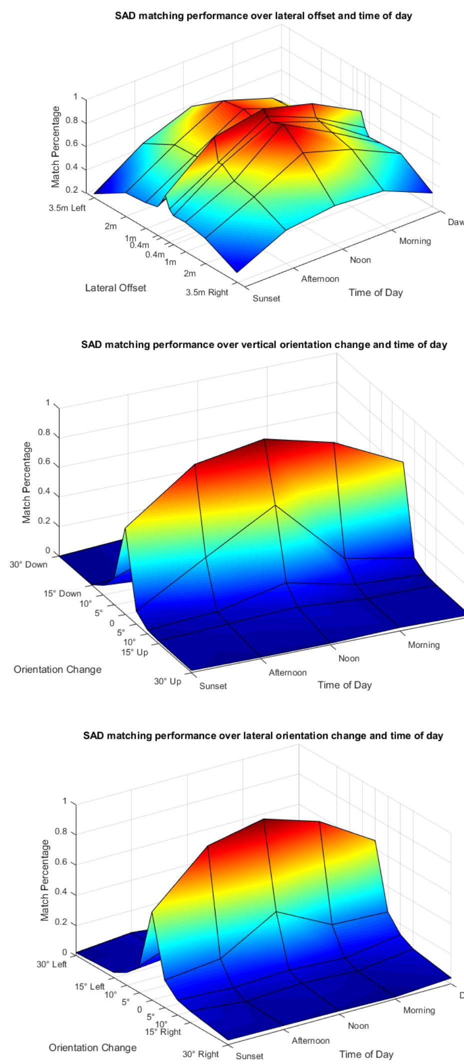

Fig. 3. Match percentage for Sum of Absolute Differences for, from top to bottom, lateral offset, vertical orientation change (pitch), and lateral orientation change (yaw). Each offset was tested across several times of day.

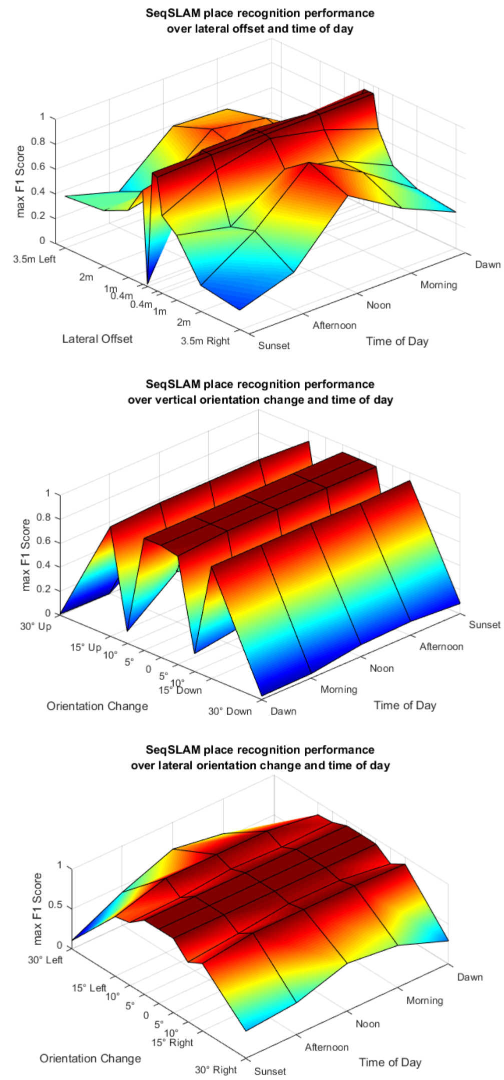

Fig. 4. SeqSLAM performance over, from top to bottom, lateral offset, vertical orientation change (pitch), and horizontal orientation change (yaw). Each viewpoint change is captured over 5 times of day to evaluate how performance changes over both viewpoint and condition change

performance cases. The corresponding input parameters of SeqSLAM which might have caused the unexpected intermittent performance drop as shown in the Figure 4 could be the *Offset Matching Range* and/or *Patch Normalization* factor. We further analysed the performance drop for different times of day at $10°$ pitch. Figure 7 shows the SAD matrix comparison with ground truth, true positives and false positives marked. The difference matrices show that the poor performance is being introduced by multiple aliased matches, suggesting that this particular combination of environmental parameters increases the aliased nature of the environment.

F. Comparison to real-world

It is important to verify that performance change in simulation is representative of performance change in the real world. To test this, we compared performance drop over lateral viewpoint change using SeqSLAM for data from a real street and for data generated from a simulation of a similar street. The real data experiment images are shown in Figure 8 and the performance results can be seen in Figure 9.

Unsurprisingly, observed performance is consistently better in simulation; we ascribe this to a lack of sufficient realism in the simulation, which was built simply and quickly. However, the performance follows the same trends in both environments, falling off smoothly as the offset difference increases. This demonstrates that performance results obtained in simulation obey similar trends to data obtained from even simplistic simulations.

IV. Analysis of Object Recognition Viewpoint Dependency

Another area of robotics research that can benefit from high fidelity simulation is visual object recognition and detection. Much of the recent progress in this field has

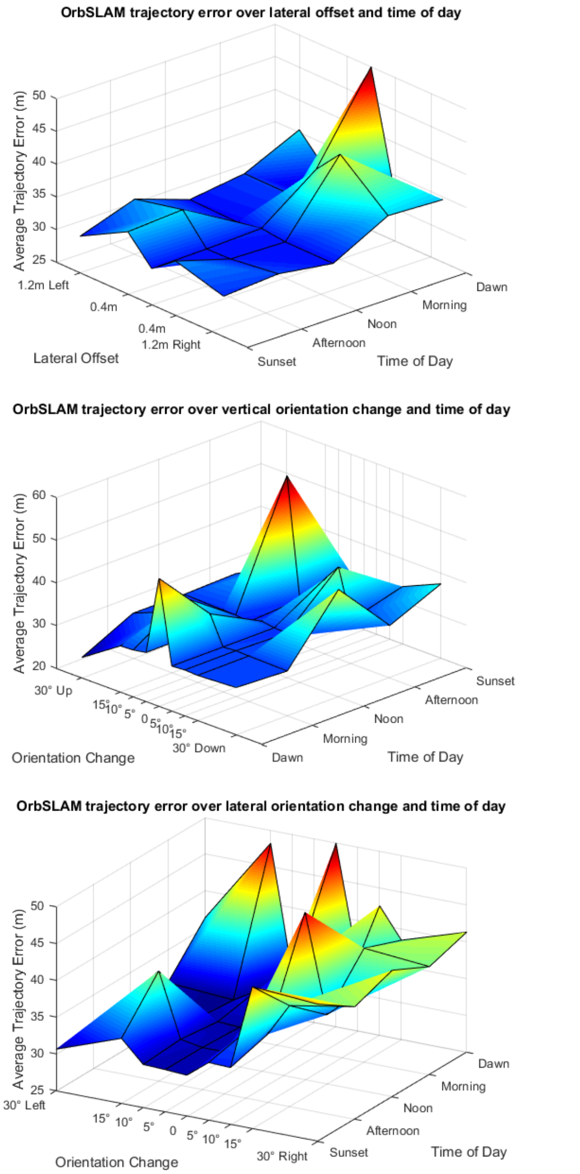

Fig. 5. ORB-SLAM average trajectory error over, from top to bottom, lateral offset, vertical orientation change (pitch), and horizontal orientation change (yaw). Each viewpoint change is captured for 5 different times of day.

been driven by the computer vision community's intensive race to achieve ever improving performance on large image recognition datasets like ImageNet [21] that is curated from large online photo repositories. State-of-the-art deep learning techniques now rival or surpass human performance and often approach perfect performance on these test sets. Rationally, related research fields such as robotics should be the beneficiary of such significant advances, but with a few exceptions, deep learning techniques have made little headway into the robotics field. Perhaps the primary reason for this disconnect is that robots operating in unpredictable, real world environments encounter visual imagery with very different characteristics and biases (or lack of biases) to that seen in traditional computer vision datasets.

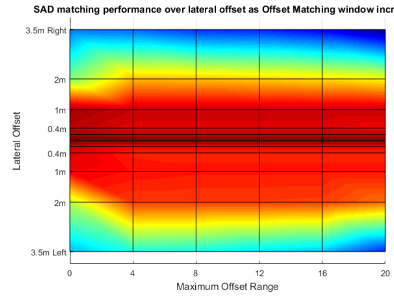

Fig. 6. Sum of Absolute Difference matching rate as the maximum offset window is increased. This result helps us choose a maximum offset appropriate for a given lateral offset

The high-fidelity simulation framework discussed in this paper allows to render realistic views of arbitrary objects from different viewpoints, under different lighting conditions, backgrounds, and occlusions. Such a dataset enables an in-depth analysis of current object recognition approaches to understand under which conditions and viewpoints good performance can be expected, and to inform and improve the training process to address the discovered challenges.

To demonstrate the potential of such analysis, we perform a viewpoint dependency test for an exemplary state-of-the-art convolutional network for object recognition. We use the simulation framework to generate views of a coffee mug object by moving the camera around the object on a sphere. We increment pitch and yaw in 5 degrees steps while keeping the camera pointed at the object centre. The generated images were then classified by the `vgg_s` convolutional network [3]. Fig. 10 illustrates the results and reveals the object-specific viewpoint dependency: The mug could only be correctly identified when the handle was clearly visible, sticking out to the side. Furthermore, even with the handle visible, the classification failed under insufficient lighting conditions (e.g. in the area around pitch 175, yaw 100). Interestingly the mug can be identified correctly if seen from above, i.e. looking inside the mug (pitch 175 degrees), but not when observing the mug from below. Such insights can help improve the training process of object classifiers and the simulation framework can even be used to generate more training data from viewpoints or under conditions that are hard to replicate in reality.

V. DISCUSSION

The most difficult and time consuming aspect of using high-fidelity simulation is creating the scene in the first place. Scenes are built of 3D models and textures, which are time consuming to create and arrange in the scene. The more realistic the scene requirement, the more labour is required. The creation of high quality 3D models and scenes is the speciality of a professional 3D artist, so if a particular problem requires large or realistic scene it may be necessary to hire a 3D artist to construct it. It can also be extremely difficult to maintain a consistent scale throughout the scene. Examination of our scene in Figure 1 will reveal scale flaws,

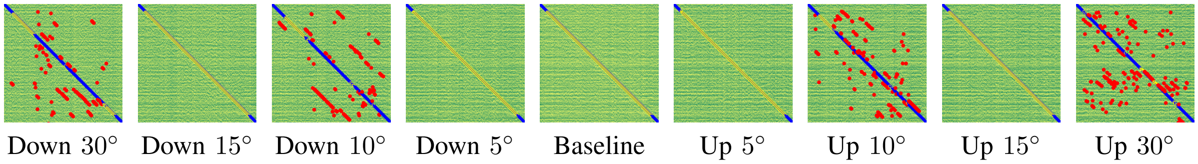

Fig. 7. The SAD matrix for comparison of Noon Baseline dataset with a change in orientation of camera's pitch. The ground truth is marked in blue and forms the diagonal of the matrix with true positives overlaid in yellow on the diagonal. The false positives are marked in red. The simulation results enable us to produce these specific confusion matrices, which clearly show that the abnormal performance is caused by increased environmental aliasing under these particular conditions, shown by the off diagonal false positive matches.

Fig. 8. Images for five different traversals of a street with different lateral offsets. This real world data (top) is used to compare the trend of performance change with change in lateral shifts as compared to the simulated data (bottom).

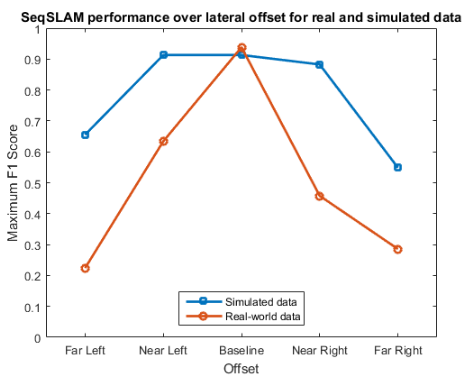

Fig. 9. Comparison of SeqSLAM performance falloff between simulated and real-world environments as lateral offset increases. As is often the case, simulated performance is better in an absolute sense, but the trend is the same in both cases.

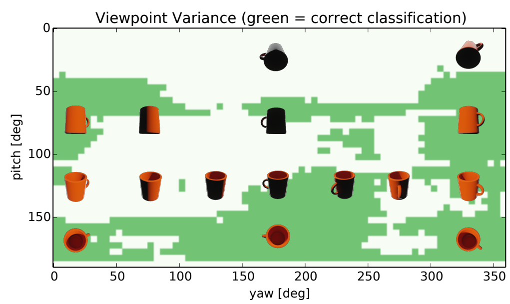

Fig. 10. Simulated viewpoint-dependency test for the `vgg_s` convolutional network [3] on a coffee mug object. Successful recognition (shaded green) requires the mug handle to be clearly visible and the object to be well-illuminated.

note for instance the height of the houses relative to the width of the road.

However, once a particular simulation has been created, the payoff is huge as it can be used to generate arbitrary amounts of image data. Once we constructed the street scene and set up the required tools, capturing each dataset could be specified programatically and takes relatively little (human) time. Indeed, when we initially created the datasets used to test the place recognition algorithms above, we tested orientation change at 30° and 15° only. However, after observing the way matching performance falls off with angle in Figure 3, we were able to very easily add tests for 5° and 10° orientation changes as well. The very fact that we can exactly repeat a movement through a scene with a precise orientation change is a powerful advantage of high-fidelity simulation.

It can in some sense be too easy to capture data. Since capturing additional data is often simply a matter of adding a new modifier to a path, increasing a sample range or decreasing a sample increment, it can be very easy to generate too much data. For instance, it may be tempting to sample a street scene such as ours at every offset up to 4m either side in 1m increments, and at each location take all vertical and horizontal changes up to 30° in 5° intervals. This is relatively simple to specify, but when multiplied out, produces $9 \times 13 \times 13 = 1521$ images per forward step down the path. The path we used with a step distance of 1m as well requires 612 forward steps, for a total therefore of 1965132 images. When generating datasets, we averaged about 6 frames per second, so generating this data would take approximately 91 hours. Further tweaking any of the specified numbers could multiply this number even further.

Rather than sampling densely with a large initial dataset, we recommend initial testing be done relatively sparsely over the test domain, with the initial results used to choose a second round of test values. For instance, given the distribution for vertical orientation change in Figure 3, it makes more sense to generate new test data at 2.5° and at 7.5° than at 25°. In the future, it should be possible to automate this iterative testing, automatically choosing new test datasets based on the results of previous testing.

It is also important to note that the difficulty of a particular change to the simulation can be non-intuitive. For instance, it is very easy in the simulation to change the location or orientation of an object or the camera; or to change the base colour of a flat-coloured object. For this reason, it was very easy for us to add additional variations on the camera path, since this simply changes the camera's location and orientation. On the other hand, changing the lighting is simple manually, but for good quality lighting a lot of data needs to be recalculated, which takes a lot of time and is not designed to be triggered programatically. As such, it takes longer to generate data across different times of day, and it would be more time-consuming for us to test at an additional time of day.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we investigated the use of photo-realistic gaming engine simulation tool to address important challenges in robotic vision across three example domains of place recognition, SLAM and object recognition: that experiments cannot be repeated exactly and that algorithms cannot be quantitatively compared. The simulation allows us to create high realistic images from an arbitrary camera or cameras in a complex 3-dimensional world in which lighting and atmospheric conditions can be fully controlled. We showed how we can systematically evaluate standard robotic vision algorithms (robust place recognition and visual SLAM) in ways which have not been previously possible. These algorithms are broadly representive of the classes of image-based and feature-based methods. We also showed how we can synthetically generate a large number of images to characterise the performance of deep networks against viewpoint change.

We have only just begun to scratch the surface of this new approach to robotic vision. So far our paths through the environment are set manually rather than being driven by a robotic vision algorithm. Integrating the game engine into the ROS environment would enable us to run a robotic vision algorithm in a manner analagous to a hardware-in-the-loop simulator, with its output commanding the camera pose in the simulator and the rendered image serving as input. We also plan to extend our testing of simulation as a tool beyond the three domains presented here.

Creating complex 3D worlds is time consuming and we are investigating modern 3D reconstruction techniques to create a first draft of a simulated world. It should also be possible to procedurally generate many parts of the environment, establishing rules for the generation of roads, cities, and other environments allowing variations to be produced quickly. Finally, many assets need only be created once, and as the community grows, the cost of setting up and using high fidelity simulations should progressively decrease.

REFERENCES

[1] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *Computer Vision–ECCV 2012*, pages 611–625. Springer, 2012.

[2] Stefano Carpin, Mike Lewis, Jijun Wang, Stephen Balakirsky, and Chris Scrapper. USARSim: A robot simulator for research and education, 2007.

[3] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *Proc. of British Machine Vision Conference (BMVC)*, 2014.

[4] M. Cummins and P. Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *The International Journal of Robotics Research*, 30(9):1100–1123, nov 2010.

[5] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014.

[6] Brian P Gerkey, Richard T Vaughan, and Andrew Howard. The Player / Stage Project : Tools for Multi-Robot and Distributed Sensor Systems. *Proceedings of the International Conference on Advanced Robotics (ICAR 2003)*, (Icar):317–323, 2003.

[7] Ankish Handa, Thomas Whelan, John McDonald, and Andrew J Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *Robotics and automation (ICRA), 2014 IEEE international conference on*, pages 1524–1531. IEEE, 2014.

[8] Brian Karis and Epic Games. Real shading in unreal engine 4. *part of Physically Based Shading in Theory and Practice, SIGGRAPH*, 2013.

[9] N. Koenig and a. Howard. Design and use paradigms for Gazebo, an open-source multi-robot simulator. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, 3:2149–2154, 2004.

[10] William A Mattingly, Dar-jen Chang, Richard Paris, Neil Smith, John Blevins, and Ming Ouyang. Robot design using unity for computer games and robotic simulations. In *2012 17th International Conference on Computer Games (CGAMES)*, pages 56–59. IEEE, 2012.

[11] Colin McManus, Ben Upcroft, and Paul Newmann. Scene signatures: Localised and point-less features for localisation. 2014.

[12] Michael Milford and Gordon Wyeth. Persistent navigation and mapping using a biologically inspired slam system. *The International Journal of Robotics Research*, 29(9):1131–1153, 2010.

[13] Michael J Milford and Gordon F Wyeth. Mapping a Suburb With a Single Camera Using a Biologically Inspired SLAM System. *IEEE Transactions on Robotics*, 24(5):1038–1053, 2008.

[14] Michael J. Milford and Gordon F. Wyeth. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1643–1649, 2012.

[15] Raul Mur-Artal, JMM Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *Robotics, IEEE Transactions on*, 31(5):1147–1163, 2015.

[16] Richard A. Newcombe, Steven J. Lovegrove, and Andrew J. Davison. DTAM: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*, pages 2320–2327. IEEE, nov 2011.

[17] S Niko, Sareh Shirazi, Adam Jacobson, Feras Dayoub, Edward Pepperell, Ben Upcroft, and Michael Milford. Place Recognition with ConvNet Landmarks: Viewpoint-Robust, Condition-Robust, Training-Free. *Robotics Science and Systems*, 2015.

[18] Martin Peris, Atsuto Maki, Sara Martull, Yoshihiro Ohkawa, and Kazuhiro Fukui. Towards a simulation driven stereo vision system. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1038–1042. IEEE, 2012.

[19] Ravi Rathnam and Andreas Birk. Initial results of cooperative auv exploration in a high-fidelity simulation using real-world data from monte da guia, azores. In *OCEANS-Bergen, 2013 MTS/IEEE*, pages 1–6. IEEE, 2013.

[20] Raul Mur-Artal and Juan D. Tardos. Probabilistic Semi-Dense Mapping from Highly Accurate Feature-Based Monocular SLAM. *Proceedings of Robotics: Science and Systems, Rome, Italy*, 2015.

[21] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2014.

[22] N. Sunderhauf and P. Protzel. BRIEF-Gist - Closing the loop by simple means. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1234–1241. IEEE, sep 2011.