Accurate, Low-Latency Visual Perception for Autonomous Racing: Challenges, Mechanisms, and Practical Solutions

Kieran Strobel[1], Sibo Zhu[2], Raphael Chang[1], and Skanda Koppula[3]

***Abstract*— Autonomous racing provides the opportunity to test safety-critical perception pipelines at their limit. This paper describes the practical challenges and solutions to applying state-of-the-art computer vision algorithms to build a low-latency, high-accuracy perception system for DUT18 Driverless (DUT18D), a 4WD electric race car with podium finishes at all Formula Driverless competitions for which it raced. The key components of DUT18D include YOLOv3-based object detection, pose estimation and time synchronization on its dual stereovision/monovision camera setup. We highlight modifications required to adapt perception CNNs to racing domains, improvements to loss functions used for pose estimation, and methodologies for sub-microsecond camera synchronization among other improvements. We perform an extensive experimental evaluation of the system, demonstrating its accuracy and low-latency in real-world racing scenarios.**

I. INTRODUCTION

Autonomous racing presents a unique opportunity to test commonly-applied, safety-critical perception, and autonomy algorithms in extreme situations at the limit of vehicle handling. As such, work on autonomous race cars has seen a strong uptick since the 2004 DARPA Challenge [33], from both industry [4, 6] and academia [13, 16, 17, 27, 34, 39]. In recent years, autonomous vehicles have grown increasingly reliant on camera-based computer vision perception [9, 19], due to the sensor modality's low cost and high information density. This paper contributes a full-stack design used to translate state-of-the-art computer vision algorithms into an accurate, low-latency computer vision system used on DUT18 Driverless, a 4WD electric racecar with podium finishes at all Formula Driverless competitions in which it raced [1, 2, 3].

Of critical importance in autonomous driving is the latency of the hardware and software stack. Latency directly impacts the safety and responsiveness of autonomous vehicles. Based on measurements from DUT18D, the visual perception system in an autonomous vehicle dominates the latency of the entire autonomy stack (perception, mapping, state estimation, planning, and controls). In particular, we find that perception occupies up to 60% of the end-to-end latency. Prior work corroborates this observation citing the intense computational load of modern visual processing as the source of latency bottlenecks in high-speed autonomous vehicles [11]. Despite its importance, low-latency visual perception of environment landmarks remains riddled with practical challenges across the entire stack, from noisy image capture and data transmission to accurate positioning in an unmapped environment. To our knowledge, there remains no available prior work detailing the full-stack design of a high-accuracy, a low-latency perception system for autonomous driving.

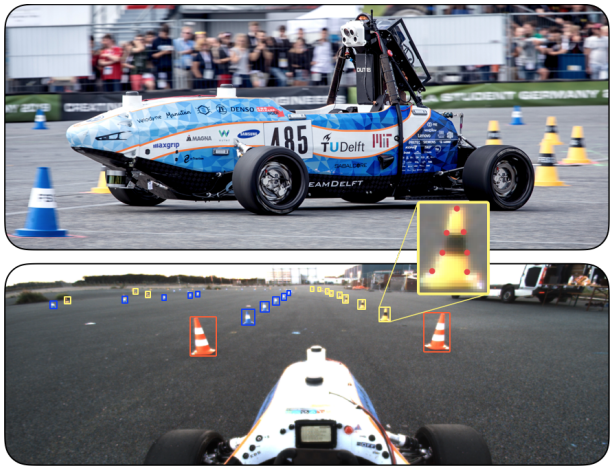

Fig. 1. DUT18D racing on track, with vision hardware mounted on the car's main roll hoop (top). Frame captured from the monocular camera (bottom), with an example detected track landmark (inset)

To this end, we describe the design and evaluation of the entire visual perception system employed by DUT18 Driverless. DUT18D is an autonomous racecar boasting straight-line acceleration of 1.4G and top-speeds of over 100kph. The racecar was built for the Formula Student Driverless competitions [12]. The visual perception system on DUT18D was designed to perceive and position landmarks on a map using multiple CNNs for object detection and depth-estimation. The design targeted high-accuracy and low-latency performance across outdoor weather and lighting conditions encountered in Boston, Italy, Germany, and the Netherlands.

This paper provides four key contributions: (i) an open design and evaluation of a thoroughly-tested low-latency vision stack for high-performance autonomous racing, (ii) solutions to implementation bottlenecks deploying state-of-the-art CV algorithms: new techniques for domain adaptation of pre-trained CNN-based object detectors, useful loss function modifications for landmark pose estimation, and microsecond time synchronization of multiple cameras, (iii) open-source C++ modules for mobile-GPU accelerated ONNX-DNN inference, landmark pose estimation, and a complete plug-and-play visual perception system for Formula Student racecars[4], and (iv) a publicly available 10K+ pose-estimation/bounding-box dataset for traffic cones of multiple colors and sizes[4].

[1] MIT, Cambridge, MA 02139 kstrobel, raphc@mit.edu
[2] Brandeis University, Waltham, MA 02453 siboz@brandeis.edu
[3] Google DeepMind, London, UK N1C 4AG skandak@google.com
[4] Open-source repositories/data: driverless.mit.edu/cv-core
Work supported MIT/DUT Racing sponsors : driverless.mit.edu

II. RELATED WORK

Prior work on autonomous racing has largely focused on full-vehicle integration [6, 34], multi-sensor fusion [13], controls [5, 16, 19, 27], and planning [38, 39], whereas this work focuses on the development of a visual perception system in this domain. Other prior works develop effective CNNs for individual vision tasks, e.g, object detection [15, 22, 30, 31, 37], monocular depth estimation [24, 25, 35], semantic segmentation [7, 29, 32], lane-keeping [8], and drone obstacle detection [20, 21, 23]. While these works provide guidance on CNN design, we find the many stop short after using validation data to evaluate on a mobile CPU/GPU. Our research builds on these works, translating and improving their ideas to build a complete vision stack, contributing open, validated software and perception hardware for autonomous racing.

III. SYSTEM TASK

Similar to most perception systems for autonomous racing, the goal of our perception system is to accurately localize environment landmarks (traffic cones) that demarcate the racetrack. The track is delineated by blue cones on the left, yellow cones on the right, and orange cones at the start and finish. Downstream mapping and planning systems use these landmarks to create and update the track map with a sample illustrated in Figure 2. Our perception system adheres to regulations set by Formula Student Driverless [12]. Formula Driverless competitions provided us the opportunity to validate our perception system in one of the premier autonomous racing leagues.

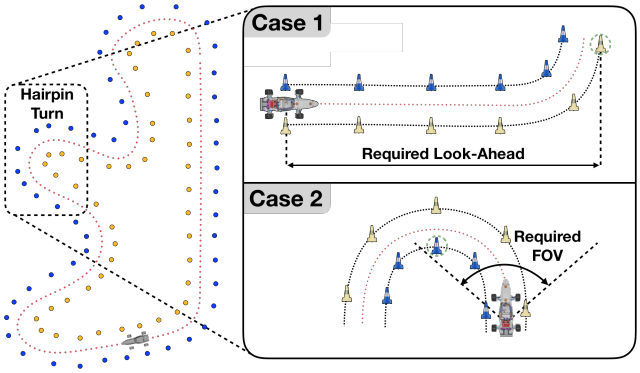

Fig. 2. Sample autocross track (left). Cases that drive minimum look-ahead requirements (top right) and FOV requirements (bottom right).

IV. SYSTEM REQUIREMENTS

Our perception system was designed to meet four high-level requirements:

1) *Mapping Accuracy*: accuracy of landmark localization.
2) *Latency*: total time between a landmark coming into view of the perception system to the time at which it is localized.
3) *Look-ahead Distance*: longest straight-line distance in which accuracy is maintained.
4) *Horizontal Field-of-View (FOV)*: arc of visibility in front of the car, related to visibility through a hairpin.

To guide design choices, we derived quantitative targets for each requirement. *Mapping accuracy* was driven by the error tolerances of downstream mapping and state estimation algorithms [14, 26]. This dictated a maximum tolerable localization error of $<$0.5m at the maximum look-ahead distance. For *latency*, we developed a kinematics model of DUT18D and simulated a safe, emergency stop from the point of top speed (25m/s) during a Formula Student acceleration run. From this, we derived a maximum 350ms view-to-actuation latency, of which downstream systems required 150ms. This translated to a latency budget of 200ms. *Horizontal FOV* is lower-bounded by unmapped hairpin turns (Figure 2). In such turns, the system must perceive landmarks on the inside apex of a hairpin turn from the start of the turn in order to plan an optimal trajectory. Given legal track dimensions, this results in a minimum FOV of 101°.

Look-ahead requirements depend on full-stack-latency, car dynamics, and camera properties. In particular, minimum landmark size, the number of pixels required to detect a landmark, plays a crucial role in determining look-ahead. To understand this more clearly, we built camera and kinematics models to generate Figure 3, a characterization of the relationship between the minimal landmark size, camera focal length, and the physical size of pixels on the camera sensor. We assume a conservative minimum detectable landmark size of 20 pixels (consistent with state-of-art work [10, 30]); this is represented by the dashed line in Figure 3. The optimal solution is a system that (1) stays close but above this line, (2) maximizes pixel size (i.e., maximizing light capture), and (3) minimizes focal length (i.e., maximizing FOV). Figure 3 illustrates the trade-off between these properties. Based on these models and fixed values of prior constraints, we derived a look-ahead requirement of 19.6m.

As there is an inherent trade-off between look-ahead distance and FOV in camera-based perception systems, the two cases outlined in Figure 2 have mutually exclusive requirements: wide FOV while maintaining long look-ahead. This motivates the two camera solution detailed in the next section.

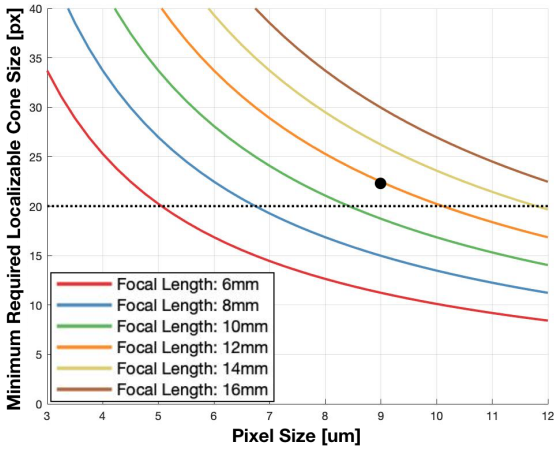

Fig. 3. Long-range camera options. The chosen configuration point (9.1um pixels/12mm focal length) is plotted for reference.

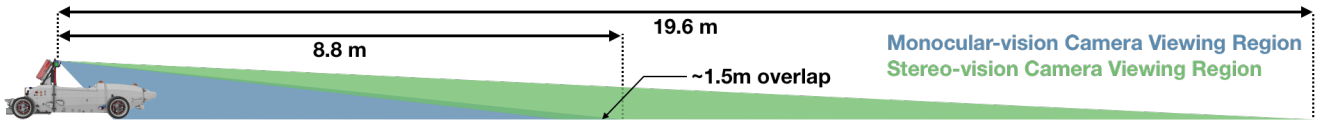

Fig. 4. Camera Viewing Regions (Side View)

V. HARDWARE OVERVIEW

To meet the requirements outlined in Section III and protect for each of the two corner cases, a two-camera architecture was deployed with a stereo pair used for long-range detections and a monocular camera for short-range detections. The camera system was mounted within the roll hoop of the vehicle, which satisfied constraints from the competition rules and allowed the cameras to be as high as possible to limit the effects of occlusions between landmarks.

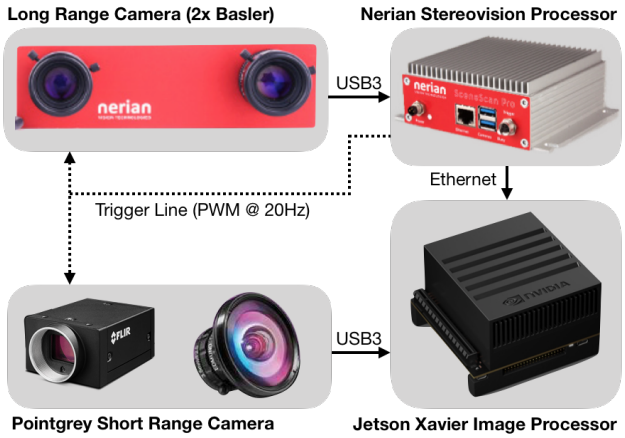

Fig. 5. Perception System Hardware Overview

Although depth estimation through stereovision is inherently less complex than using monocular vision, a dual stereovision system was infeasible due to packaging constraints, forcing the use of a monocular vision solution for one of the two systems. The rationale for using the monocular camera for short-range rather than long-range detections is that for a reasonable mounting height, a landmark's 3D location on a relatively flat surface is a much stronger function of pixel space location for short-range objects than long-range objects. This relieves some of the challenges for estimating landmark pose from a monocular camera.

For the stereo pair, two Basler daA1600-60um sensors with 4.55um pixels (2x sensor binned) were accompanied by 12mm Hikvision lenses separated by a 10cm baseline. The monocular system consisted of a Pointgrey Grasshopper3 GS3-U3-23S6C-C with 5.9um pixels and a 3.5mm Edmund wide-angle lens. The cameras were angled downwards and shared a 1.5m overlap region to protect for mounting tolerances resulting in the viewing regions shown in Figure 4. The images were cropped to remove pixels above the horizon to reduce computational load during object detection and in turn produce a lower latency system.

The stereo pair is connected directly to a stereo matching FPGA supplied by Nerian Vision Technologies which creates low latency dense disparity maps. These disparity maps, along with images from both pipelines, are then processed on an Nvidia Jetson Xavier embedded GPU where landmarks are detected in the images, and depth estimates are extracted. The localized landmarks are then sent to a separate compute unit for the rest of the autonomous pipeline. An overview of the hardware used for this system is shown in Figure 5.

VI. SOFTWARE STACK

To produce pose estimates of landmarks from raw images, the pipeline follows three logical steps:

1) Data Acquisition: Synchronized image streams are captured, disparity matched for the stereovision pipeline, and transferred to the Jetson Xavier. A critical component of this is time synchronizing all devices.
2) 2D Space Localization: Using a neural network-based approach, landmarks are detected and outlined by bounding boxes in the images.
3) 3D Space Localization: For the stereovision pipeline, depth from each landmark is extracted by a clustering-based approach. A neural network-based approach is used to compute depth from the monocular camera. A block diagram each pipeline is shown in Figure 6.

A. Data Acquisition

For high-speed autonomous vehicles to accurately localize landmarks, precise time stamps must accompany each image in a common time base. Without this, inaccuracies as small as 10ms can result in 30cm errors at high speeds – over 50%

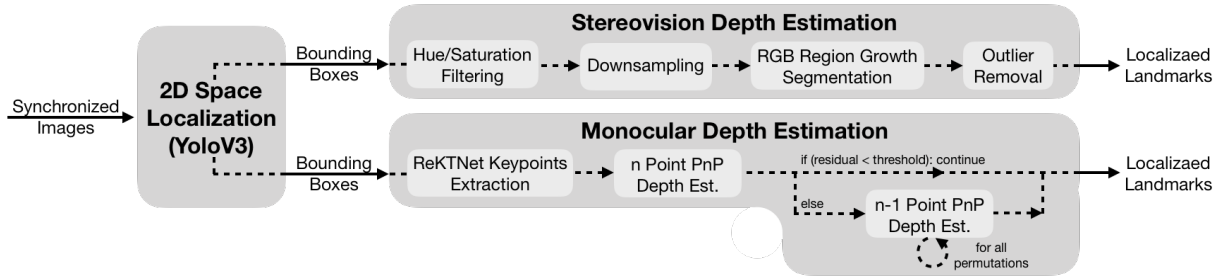

Fig. 6. Depth Estimation Pipeline Overview

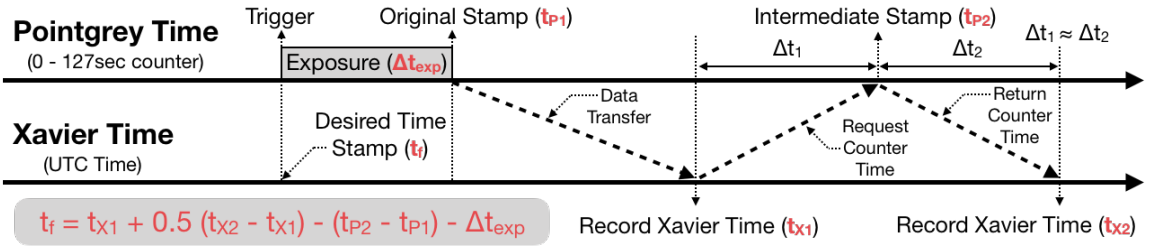

Fig. 7. Monocular camera time synchronization. Variables used for time-translation calculations are highlighted in red for clarity.

of the maximum localization error budget. To solve this, we developed a system in which both pipelines are triggered by a hardware timestamped signal generated by the Nerian FPGA which is synchronized using the IEEE PTP protocol to Xavier's master clock. This results in sub-microsecond accurate timestamps, translating to sub-millimeter localization errors contributed by the time synchronization. For the USB3 monocular camera, a custom time-translation process was developed, as shown in Figure 7. As the monocular camera is triggered off of the same signal as the stereovision system, the timestamp (t_f) calculated from this process is then re-stamped with the closest stereovision image timestamp, resulting in sub-microsecond synchronization over USB3. Adjustments above 10ms are rejected to ensure that dropped frames do not corrupt this process.

In the case of the monocular vision pipeline, there is an inherent trade-off between localization accuracy and latency. Methods for monocular depth estimation outlined in Section II require high-resolution images, whereas the latency of CNN based object detection algorithms scale linearly as resolution increases. To circumnavigate this, pictures from our monocular camera are captured at 1600x640 and transferred to the Jetson Xavier where they are later software binned to 800x320. The low-resolution images are used for object detection, where the high-resolution images are used for depth estimation. This allows for high accuracy depth estimation while maintaining low-latency object detection. As the system performance is bottlenecked by the accuracy of the monocular depth estimation process, using low-resolution images for object detection does not result in a performance reduction. Images from the stereovision pipeline are also captured at 1600x640 but binned on the camera's ISP before being transferred to the Nerian FPGA. Hardware binning was done to increase the amount of light captured by each pixel, as the stereo matching process met all requirements without the need for higher resolution. A semi-global matching algorithm is run on the FPGA to produce a dense disparity map with a minimum depth of 4.2m. The disparity map and one image from the stereo pair are then transferred via Ethernet to the Xavier.

B. 2D Localization: YOLOv3

For accurate 2D localization, the synchronized images are batched together and passed through a full YOLOv3 [30] neural network using the TensorRT inference framework. Weights were calibrated to int-8 precision using 40k pictures from the network training dataset resulting in inference speeds 10x faster than a similar PyTorch implementation. Non-maximal suppression thresholds were set to 10% to filter out occluded landmarks as these were found to be problematic for depth estimation later in the pipeline.

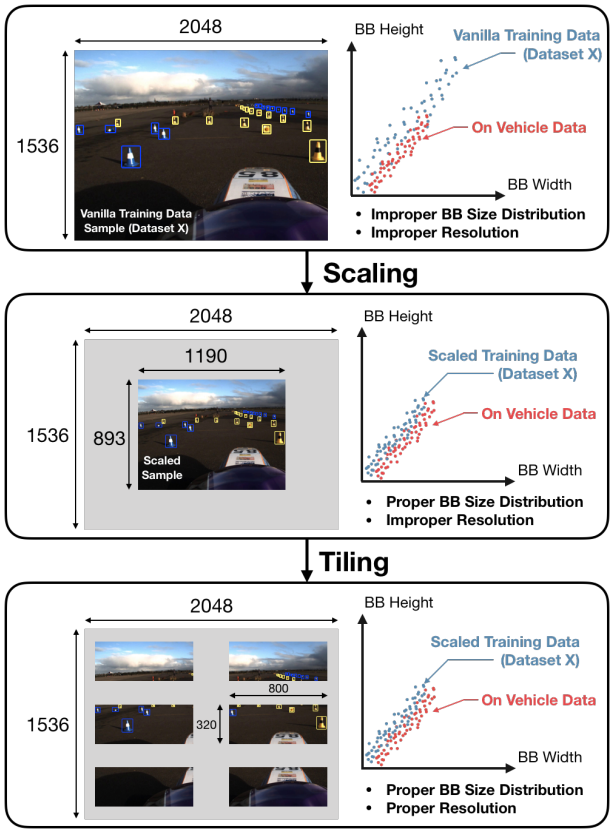

Fig. 8. CV-C Dataloader Pre-processing Stages

A perception neural network that can generalize across different domains (weather, lighting, scenery) will produce more robust detections and in turn localize landmarks with higher accuracy. To ensure this with our networks, training data was collected on multiple image sensors and lenses from various perspectives in different settings. A drawback of this process is that the distribution of landmark bounding box (BB) sizes (in pixels) in the training set no longer was representative of what would be seen by the network in the wild. To mitigate this, each set of training images from a specific sensor/lens/perspective combination was uniformly rescaled such that their landmark size distributions matched that of the camera system on the vehicle. Each training image was then padded if too small or split up into multiple images

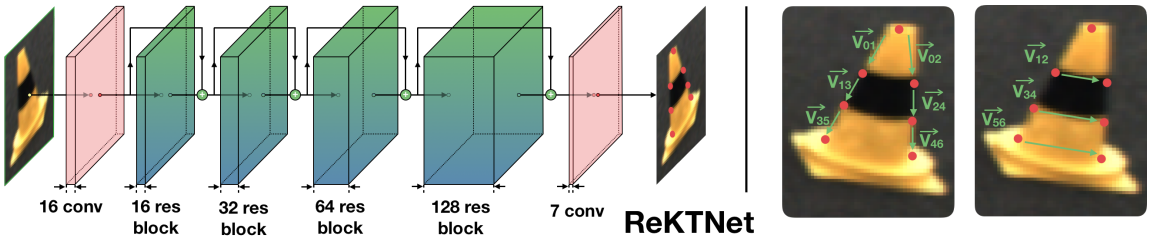

Fig. 9. ReKTNet architecture (left). Vectors used in geometric loss function (right)

if too large. This process is illustrated in Figure 8. Since YOLOv3 makes width and height predictions of detections by resizing predefined bounding boxes, k-means clustering was done on the post-scaled training data in height and width space to give the network strong priors.

An additional modification made during the training process was tuning the hyperparameters in front of each of the terms in the loss function. Each bounding box prediction consisted of estimates at x, y, width, height, class (foreground or background), and confidence, which are penalized differently during training as follows:

$$L_{total} = \gamma_{cls}L_{cls-ce} + \gamma_{BG}L_{BG-bce} + \gamma_{FG}L_{FG-bce} + \gamma_{xy}(L_{x-mse} + L_{y-mse}) + \gamma_{wh}(L_{w-mse} + L_{h-mse}) \quad (1)$$

A distributed Bayesian hyperparameter search for the five coefficients resulted in significant gains in precision when weighting the foreground loss two-orders of magnitude greater than the background loss. The converged upon values from the optimization process are γ_{BG}=25.41, γ_{FG}=0.09, γ_{XY}=1.92 and γ_{WH}=1.33. Further gains were obtained by switching from the SGD optimizer in the initial implementation to Adam [18]. The compounding benefits for each of these changes are shown in Figure 10 as precision-recall curves. The resulting final mAP was 85.1% with 87.2% recall and 86.8% precision.

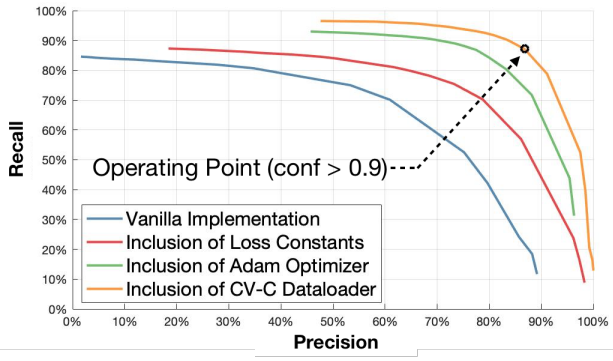

Fig. 10. Precision-Recall Curves for Compounding YOLOv3 Modifications

C. 3D Localization: Monocular Vision

For the monocular pipeline, a priori knowledge of the landmark dimensions is leveraged to provide scale to images, and in turn, extract depth estimates from a single camera. To do this, an additional residual neural network, ReKTNet, trained with 3.2k images based off of work by [9] is run to detect seven key points on each YOLO detection for use in a Perspective-n-Point (PnP) algorithm. The network architecture, along with a sample output, is shown in Figure 9. To make the algorithm robust to single keypoint outliers all subset permutations of the keypoints with one point removed are calculated if the reprojection error from the PnP estimate using all keypoints is above a threshold. The permutation with the lowest error is used as the final estimate.

Two important modifications were made to the network. First, the fully connected output layer was replaced with a convolutional layer, which predicts a probability heatmap over the image for each keypoint, and the expected value over the heatmap [28] is used as the keypoint location. The use of a fully convolutional network not only reduces the number of network parameters for faster convergence, but is also a better choice given the benefits of convolutional layers for tasks of predicting features that are spatially interrelated. The second modification is an additional term in the loss function to leverage the geometric relationship between points. Since the keypoints on the sides of a cone are collinear, the dot products of the unit vectors between points on these lines should be one. One minus the values of these dot products are used directly in the loss function. The same is done for the three horizontal vectors across the cone. An illustration of these vectors is shown in Figure 9. Because the keypoint locations now need to be backpropagated, the differentiable expected value function [28] is used to extract coordinates from heatmaps. The final loss function is as follows:

$$L_{total} = L_{mse} + \gamma_{horz}(2 - V_{12} \cdot V_{34} - V_{34} \cdot V_{56}) + \gamma_{vert}(4 - V_{01} \cdot V_{13} - V_{13} \cdot V_{35} - V_{02} \cdot V_{24} - V_{24} \cdot V_{46}) \quad (2)$$

Using a Bayesian optimization framework like the one previously described, the values were determined to be $\gamma_{vert} = 0.038$ and $\gamma_{horz} = 0.055$.

D. 3D Localization: Stereo Vision

To accurately localize landmarks from the stereovision pipeline, 3D points from detections are first passed through a hue and saturation filter where all pixels with H or S values below 0.3 are neglected. This not only removes the majority of the asphalt background within each bounding box but also removes the stripe on the cone from being considered in the depth estimation. This is beneficial as the stereo matching algorithm typically produces high depth variance disparity maps within this stripe due to lack of texture. To reduce the latency of each detection, the point clouds within the bounding box are downsampled if there are more than 200 points remaining. The points are then clustered based on their location in XYZ and RGB space, and the XYZ-centroid of the cluster with the closest average hue to the predicted cone color is used as the location of the cone.

VII. Validation

Validation efforts for this perception system were focused on validating the four high-level requirements of Section IV. The FOV requirement was satisfied through lens and sensor selection, resulting in a horizontal FOV of 103°.

To validate system latency, all eight cores on the Xavier CPU were artificially stressed as each pipeline node was profiled. As the depth of the landmarks is processed in series, the latency linearly depends on the number of landmarks detected. For this study, a track was artificially set up such that the system detected 20 cones in total. In practice, the given look-ahead results in fewer than 20 cones detected, especially in corner cases outlined in Section III where latency is most important, making these results conservative.

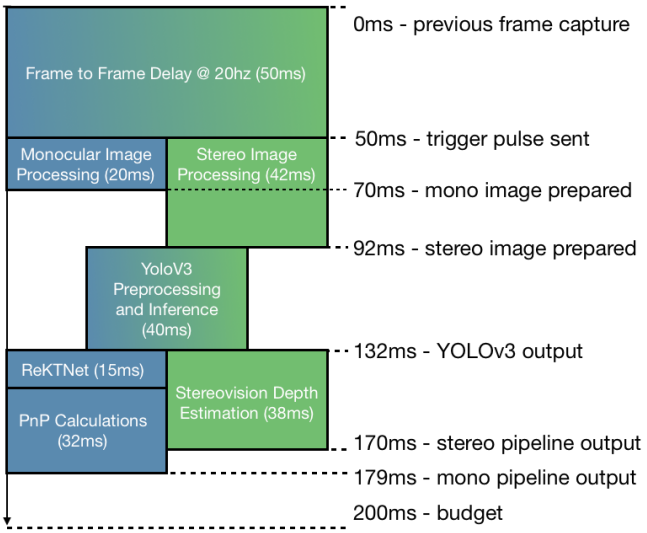

Fig. 11. Network latency. Monocular pipeline to the left, stereovision pipeline to the right.

As is clear from Figure 11, both pipelines running in parallel publish landmark locations within the budgeted perception system latency. This measurement includes the frame to frame delay, which accounts for scenarios where landmarks come into the FOV immediately after the camera shutter closes and is responsible for 28% of the total latency. This component of the latency stack is typically overlooked. To provide context for these values, the median visual perception latency of a human is 261ms for tasks far more simplistic [36] – our perception system outdoes this by 45%.

To characterize accuracy of the localization phase of the pipeline, we examined detection accuracy across three landmark sizes. The results are shown in Figure 12. We use a widely-used metric, intersection-over-union (IoU), that measures alignment of our bounding box with ground truth [30]. We examined IoU across >24,000 landmarks. We achieve a median IoU of 88% for large cones, and 83-84% for smaller cones. This bounding box tightness enables whole system localization accuracy described next.

The final two requirements, accuracy and look-ahead, were validated in tandem using the whole system. Using the track illustrated in Figure 2, the vehicle was statically placed at

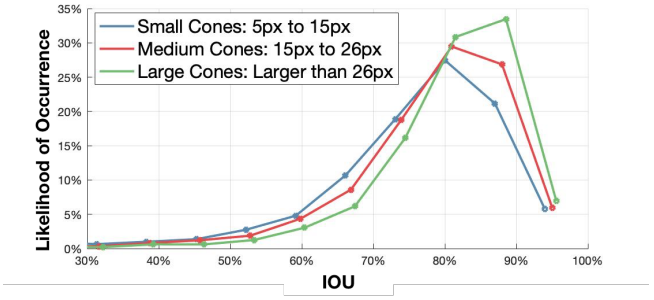

Fig. 12. Landmark Detection IOU Distribution for Various BB Sizes

five different locations, and 20 seconds of data were collected on the estimated distances of the landmarks in the system's FOV. Distance errors over various distances are shown in Figure 13. Mean errors for the 20 seconds of recorded data were below 0.5m for both pipelines, and standard deviations were below 5cm for the monocular pipeline and 10cm for the stereo pipeline. Ground truth values for Euclidean distances from the cameras were measured using a Leica Disto D1. The experiment was done statically to remove the dependency on the vehicle's state estimation system. The results reveal a maximum effective disparity offset of 0.15 pixels in the stereovision pipeline achieved through the clustering algorithm, which is 40% lower than the reported value on the datasheet.

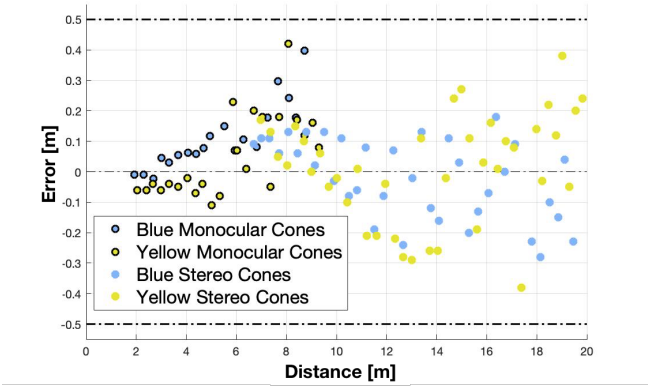

Fig. 13. Depth estimation error vs. distance. Dotted lines provide bounds set by the system requirements.

VIII. Conclusions

Accurate and low-latency visual perception is applicable to many domains, from autonomous driving to augmented reality. We present challenges that required us to optimize known solutions to develop designs optimized for Formula Driverless. The end result of this work is a perception system that is able to achieve an estimated sub-180ms latency from perception to depth estimation, with errors less than 0.5m at a distance of 20m. This system and its associated source code are available for other teams as a baseline to build off of to facilitate progress in high-performance autonomy.

Acknowledgment

We thank all the members, advisors, and generous sponsors of DUT/MIT Racing for making this project possible.

REFERENCES

[1] *Autonomous Driving at Formula Student Germany*. https://www.formulastudent.de/pr/news/details/article/autonomous-driving-at-formula-student-germany-2017/.

[2] *Formula Student Germany Results*. https://www.formulastudent.de/fsg/results/2019/.

[3] *Formula Student Italy Results*. https://www.formula-ata.it/results-2/.

[4] *Self Racing Cars*, Jan 2019. http://selfracingcars.com/.

[5] Michael Garrett Bechtel, Elise McEllhiney, and Heechul Yun. Deeppicar: A low-cost deep neural network-based autonomous car. *CoRR*, abs/1712.08644, 2017.

[6] Danilo Caporale, Alessandro Settimi, Federico Massa, Francesco Amerotti, Andrea Corti, Adriano Fagiolini, Massimo Guiggian, Antonio Bicchi, and Lucia Pallottino. Towards the design of robotic drivers for full-scale self-driving racing cars. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5643–5649. IEEE, 2019.

[7] Liangfu Chen, Zeng Yang, Jianjun Ma, and Zheng Luo. Driving scene perception network: Real-time joint detection, depth estimation and semantic segmentation. *CoRR*, abs/1803.03778, 2018.

[8] Z. Chen and X. Huang. End-to-end learning for lane keeping of self-driving cars. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1856–1860, June 2017.

[9] Ankit Dhall. Real-time 3d pose estimation with a monocular camera using deep learning and object priors on an autonomous racecar. *arXiv preprint arXiv:1809.10548*, 2018.

[10] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *CoRR*, abs/1406.2283, 2014.

[11] Davide Falanga, Suseong Kim, and Davide Scaramuzza. How fast is too fast? the role of perception latency in high-speed sense and avoid. *IEEE Robotics and Automation Letters*, 4(2):1884–1891, 2019.

[12] Formula Student Germany. *Formula Student Rules 2019. V1.0*.

[13] Nikhil Gosala, Andreas Bühler, Manish Prajapat, Claas Ehmke, Mehak Gupta, Ramya Sivanesan, Abel Gawel, Mark Pfeiffer, Mathias Bürki, Inkyu Sa, et al. Redundant perception and state estimation for reliable autonomous racing. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6561–6567. IEEE, 2019.

[14] Giorgio Grisetti, Gian Diego Tipaldi, Cyrill Stachniss, Wolfram Burgard, and Daniele Nardi. Fast and accurate slam with rao–blackwellized particle filters. *Robotics and Autonomous Systems*, 55(1):30–38, 2007.

[15] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[16] Juraj Kabzan, Lukas Hewing, Alexander Liniger, and Melanie N Zeilinger. Learning-based model predictive control for autonomous racing. *IEEE Robotics and Automation Letters*, 4(4):3363–3370, 2019.

[17] Juraj Kabzan, Miguel de la Iglesia Valls, Victor Reijgwart, Hubertus Franciscus Cornelis Hendrikx, Claas Ehmke, Manish Prajapat, Andreas Bühler, Nikhil Gosala, Mehak Gupta, Ramya Sivanesan, et al. Amz driverless: The full autonomous racing system. *arXiv preprint arXiv:1905.05150*, 2019.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

[19] Skanda Koppula. Learning a cnn-based end-to-end controller for a formula sae racecar. *arXiv preprint arXiv:1708.02215*, 2017.

[20] Christos Kyrkou, George Plastiras, Theocharis Theocharides, Stylianos I Venieris, and Christos-Savvas Bouganis. Dronet: Efficient convolutional neural network detector for real-time uav applications. In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 967–972. IEEE, 2018.

[21] Jangwon Lee, Jingya Wang, David Crandall, Selma Šabanović, and Geoffrey Fox. Real-time, cloud-based object detection for unmanned aerial vehicles. In *2017 First IEEE International Conference on Robotic Computing (IRC)*, pages 36–43. IEEE, 2017.

[22] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3569–3577, 2018.

[23] Seyed Majid Azimi. Shuffledet: Real-time vehicle detection network in on-board embedded uav imagery. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[24] Michele Mancini, Gabriele Costante, Paolo Valigi, and Thomas A Ciarfuglia. Fast robust monocular depth estimation for obstacle detection with fully convolutional networks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4296–4303. IEEE, 2016.

[25] Vlad-Cristian Miclea and Sergiu Nedevschi. Real-time simultaneous object detection and depth estimation.

[26] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *IJCAI*, pages 1151–1156, 2003.

[27] Jun Ni and Jibin Hu. Path following control for autonomous formula racecar: Autonomous formula student competition. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1835–1840. IEEE, 2017.

[28] Aiden Nibali, Zhen He, Stuart Morgan, and Luke Prendergast. Numerical coordinate regression with convolutional neural networks. *ArXiv*, abs/1801.07372, 2018.

[29] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *CoRR*, abs/1606.02147, 2016.

[30] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *ArXiv*, abs/1804.02767, 2018.

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[32] Ganesh Sistu, Isabelle Leang, and Senthil Yogamani. Real-time joint object detection and semantic segmentation network for automated driving. *CoRR*, abs/1901.03912, 2019.

[33] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, et al. Stanley: The robot that won the darpa grand challenge. *Journal of field Robotics*, 23(9):661–692, 2006.

[34] Miguel I Valls, Hubertus FC Hendrikx, Victor JF Reijgwart, Fabio V Meier, Inkyu Sa, Renaud Dubé, Abel Gawel, Mathias Bürki, and Roland Siegwart. Design of an autonomous racecar: Perception, state estimation and system integration. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2048–2055. IEEE, 2018.

[35] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. Fastdepth: Fast monocular depth estimation on embedded systems. *CoRR*, abs/1903.03273, 2019.

[36] David Woods, John Wyma, E. Yund, Timothy Herron, and Bruce Reed. Factors influencing the latency of simple reaction time. *Frontiers in human neuroscience*, 9:131, 03 2015.

[37] Bichen Wu, Forrest N. Iandola, Peter H. Jin, and Kurt Keutzer. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. *CoRR*, abs/1612.01051, 2016.

[38] Dean Zadok, Tom Hirshberg, Amir Biran, Kira Radinsky, and Ashish Kapoor. Explorations and lessons learned in building an autonomous formula sae car from simulations. *arXiv preprint arXiv:1905.05940*, 2019.

[39] Marcel Zeilinger, Raphael Hauk, Markus Bader, and Alexander Hofmann. Design of an autonomous race car for the formula student driverless (fsd). In *Oagm & Arw Joint Workshop*, 2017.