Ego-centric Stereo Navigation Using Stixel World

Shiyu Feng[1], Fanzhe Lyu[2], Jin Ha Hwang[3] and Patricio A. Vela[2]

Abstract— **This paper explores the use of passive, stereo sensing for vision-based navigation. The traditional approach uses dense depth algorithms, which can be computationally costly or potentially inaccurate. These drawbacks compound when including the additional computational demands associated to the sensor fusion, collision checking, and path planning modules that interpret the dense depth measurements. These problems can be avoided through the use of the stixel representation, a compact and sparse visual representation for local free-space. When integrated into a Planning in Perception Space based hierarchical navigation framework, stixels permit fast and scalable navigation for different robot geometries. Computational studies quantify the processing performance and demonstrate the favorable scaling properties over comparable dense depth methods. Navigation benchmarking demonstrates more consistent performance across high and low performance compute hardware for PiPS-based stixel navigation versus traditional hierarchical navigation.**

I. Introduction

The major advances in autonomous navigation for mobile robots are primarily due to technological achievements in active range or depth sensing coupled with improved compute hardware. These sensors support direct recovery of local scene geometry and free space estimates for planning collision avoiding trajectories. However, active depth or ranging sensors can be fooled by certain material properties, lack the sensor resolution achievable by modern RGB camera systems, and can be costly relative to cameras. Their benefits include ease of use, long sensing distances, and reduced sensitivity to the visual environment due to active illumination. In cases where the negative factors outweigh the benefits or passive sensing is required, stereo systems are candidate alternatives. As an indirect depth system, their use requires considering the computational costs of depth recovery and leads to sensing versus computing trade-offs. For mobile robots with limited on-board resources, this computational challenge requires novel ways of interpreting stereo imagery. Especially in the context of robot navigation where fast and safe decisions should be made when traversing unknown or partially known environments. Delays in stereo depth estimation may undermine collision avoidance.

An algorithmic strategy to lower runtime costs is to explore alternative representations that not only produce sparse structure data, but keep sufficient depth information to make informed decisions. One representation with algorithmic consequences that has the potential to improve the run-time properties of stereo processing for mobile ground vehicle navigation and collision avoidance is the *stixel world* [1]. A stixel compactly and efficiently represents objects in the image space much like a laser scan. It represents the ground plane location of vertical obstacles in the world plus their height, see Fig. 1(a). The colored column overlays in the image depict stixels in the world; the red to green color means near to far.

Another strategy to improve run-time is to lower the overhead of collision checking. The most common instances simplify the robot's geometric representation to point, circular, or polygonal geometries [2]. Over-simplified representations of robots cause problems when estimating safe navigable trajectories. The main source of latency and computational cost is the mixed representation between sensor data and collision geometry, and the need to make the former compatible with the latter. Recently, the idea of Planning in Perception Space (PiPS) [3] opted to reverse this process. Rather than mapping sensor data to world geometry, collision checking maps robot geometry into an ego-centric perceptual representation. PiPS reduces the level of sensory data processing required to make decisions regarding trajectory safety. Importantly it adapts to different robot geometries.

Given that stixels are sparse, compact, mid-level representations of local collision space, they are excellent candidates for use by perception-space algorithms. Augmenting the baseline stixel implementation with PiPS-compatible elements should lead to a light weight stereo navigation system. Being based on perception-space processing, the system will have favorable scaling properties as a function of the sensor resolution and the quantity of collision checks performed. These scaling properties permit the use of stixels across a diverse range of robot compute configurations, from embedded to workstation class hardware. In essence the stixel-based modifications will lead to a passive visual navigation system that 1) lowers latency without sacrificing safety in navigation, and 2) has better computational scaling.

II. Research Context

Stereo Vision. Though research into accurate stereo depth estimation algorithms is quite extensive, both Block Matching (BM) and Semi-Global Block Matching (SGBM) [1], [4] continue to be used due to easily obtained code and their ability to provide reasonable quality depth measurements from discrete disparity estimates. Relative to other methods, BM variants are relatively efficient, but relative to real-time

*This work supported in part by NSF Award #1400256 and #1849333.

[1]S. Feng is with the School of Mechanical Engineering and the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30308, USA. shiyufeng@gatech.edu

[2]F. Lyu, and P.A. Vela are with the School of Electrical and Computer Engineering and the Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30308, USA. {fanzhe, pvela}@gatech.edu

[3]J.H. Hwang is with jhwang44@mail.gatech.edu

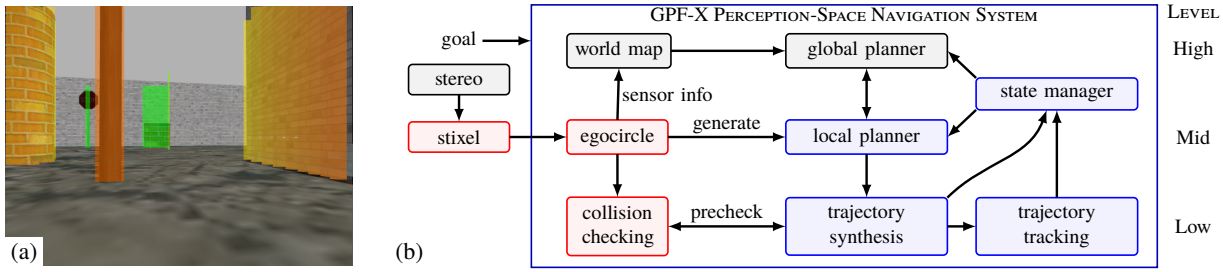

Fig. 1: (a) Stixels overlaid on a robot's camera view in a simulated environment. Stixels convert 3D geometry to 2D data that can be reduced to 1.5D data. (b) The GPF-X hierarchical navigation system with a stixel-based, perception-space processing.

navigation needs, they can be slow. Contemporary methods for dense stereo leverage the ability of deep networks and availability of GPU processors [5], [6]. Deep network approaches are tuned to the stereo configuration and to the target navigation scenario. For robots with on-board GPU's, they are good options. Likewise, if special purpose hardware can be installed, such as FPGAs, then the off-loading of the depth recovery computations both frees up compute demands and provides low latency depth maps [7], [8]. Hardware-enabled strategies for accelerating calculations also apply to stixels [9]. The primary interest is in what value the stixel representation provides.

The limiting properties of traditional stereo matching motivated the use of stixels as a compact, medium level, visual representation [1]. The intent was to use stixels for robust free space and object depth estimation from stereo cameras to support navigation decision making. To that end, stixels were shown to strengthen the priors associated to computer vision algorithms for object detection [10]–[12] and semantic segmentation [13], [14] tasks. While the original version of stixel estimation required dense depth generation as the first step, later methods directly estimated stixels in u-disparity domain and $u-v$ domain more efficiently [10], [15]. Interestingly, the stixel applications cited above are not for collision-avoiding, navigation-based front-ends though initially posited to have such a use. This paper explores the value of stixels within a vision-based hierarchical navigation system.

Vision-Based Navigation. Common navigation methods based on stereo follow the traditional perceive-plan-act pipeline, which attempts to assimilate all data into a common representation for decision-making. The most prevalent being occupancy grids, whether in 2D for mobile ground robots [16], or 3D for aerial robots [17], [18]. For large-scale navigation, these data structures are implemented in a hierarchical manner with local (small-scale) and global (large-scale) planning [19]–[21], where the small-scale representation is used for real-time path planning and sensory data integration. While doing so improves run-time performance, there are still fundamental limits and scaling issues associated to the underlying algorithms that prevent generic use across platform configurations [22]. Algorithms designed to improve the computational aspects of local planning exploit efficiencies related to point mass or spherical robot models [23], [24] and rely on advantageous scaling in the low-data throughput regime that cannot handle higher resolution sensory data [22]. Some methods also exploit the efficiency of simple linear or polygonal footprint models for collision checking [2]. More effective local planning modules for hierarchical navigation involve perception space approaches, which the stixel representation is compatible with.

Perception-Space Representations. Evidence for the value of mid-level, perception space approaches using stereo lies in the disparity space navigation method based on an egocylinder local map representation [23], [25]. Though the latency of traditional stereo matching places constraints on operational performance, the downstream perception-space processing is quite efficient [26]. A depth-based approach to perception space further showed that trajectory scoring and collision checking with non-trivial geometric models can be processed in real-time [3], [22], thereby improving compatibility with the local planners of a hierarchical navigation system [22], [27]. The results provide evidence in favor of a mixed representation hierarchical navigation system with regards to real-time performance and computational scalability. A review of the stixel representation indicates that it is a sparse visual representation mapping ground-plane contacting objects to a 1.5D polar, ego-centric, collision-space representation. Leveraging the processing framework of perception-space strategies and their local planning implementations leads to the hypothesis of this paper that: *stixel-based, passive stereo navigation will have the necessary scaling to operate on hardware constrained systems with comparable performance as on systems with better computational capabilities.*

III. Stixel Ego-centric Perception

This section describes the integration of stixel-based measurement with the core components of PiPS-based local navigation systems, the red blocks in Fig. 1(b). Before doing so, it first summarizes the stixel measurement process, and reports the outcomes of comparative studies between stixels and block matching methods regarding latency and local collision-space recovery. Once confirmed useful for navigation, stixel-based perception space modules are described. These modules include the storage and propagation of stixel measurements in time, as needed by local planner modules. It also describes how stixel measurements are compatible with perception-space collision checking and provides timing comparisons for critical processes using stixels. The timing results show that stixels have the necessary scalability properties for deployment on diverse compute platforms.

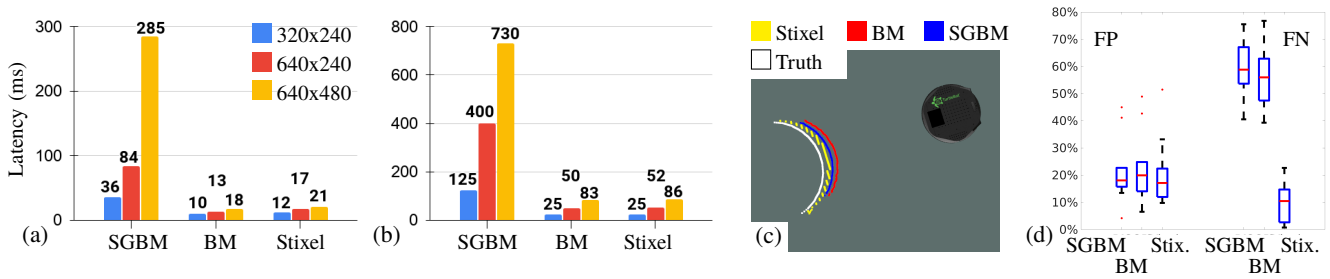

Fig. 2: Compute time and estimation accuracy. Plots the latency for different stereo depth methods and input resolutions for (a) a workstation, and (b) an embedded device. (c) Depicts the recovered depth measuremens for BM, SGBM and stixel, which serves to visualize their *safety* properties. The ranges are offset from each other for visibility. Scan completeness is important since missing measurements (false negatives) will incorrectly predict free-space. (d) Boxplot of tests measuring what percent of depth estimates are false positive/negatives. Stixel estimation captures more of the scene (less false negatives).

A. Stixel Estimation

Stixels generation presumes a calibrated stereo system whose optical axes are parallel to the horizon. It uses the relatively fast and robust stixel method u/v-disparity approach [10]. There is an initial ground plane estimation pass based on the v-disparity, followed by a stixel estimation using the u-disparity. Additional processing to recover the stixel height is not performed, since object or obstacle footprints provide sufficient scene structure for navigation by ground vehicles. Doing so reduces the stixel representation from a typically 2.5D area-based representation to a 1.5D sparse, linear one.

1) Computational Performance: To confirm the computational value of stixels with respect to the stereo-to-depth process, this section describes latency benchmarking relative to two baselines: Semi-Global Block Matching (SGBM) and Blocking Matching (BM). A simulated robot (using ROS/Gazebo) is placed in different environments and tasked to generate a depth measurement for different input resolutions and for two different compute platforms using a single-thread. The platforms are an Intel Xeon E5-2680 workstation (single-thread Passmark score: 1547) and an Atomic Pi Intel Atom x5-Z8350 (Passmark: 490). The results in Fig. 2 show that SGBM has the highest latency on both devices, and gets close to $1s$ on the Atomic Pi, which means it is impossible to support real-time sensing. BM has the lowest latencies across the board, with stixel being a few milliseconds behind (2-4.2ms).

2) Depth Estimation Performance: Though it might seem like BM and stixel are comparable, they have different properties regarding depth accuracy. For navigation, errors in the depth map compromises safety, e.g., collision avoidance. The previous experiments support measurement of *safety* levels for the depth estimation methods. The placed robot also measured the local scene using a simulated laser scanner to serve as ground truth. Mapping the depth measurements to equivalent 2D laser scan measurements permits comparison with the laser scanner. Picking a depth error percentange tolerance ($\tau_{depth} = 3\%$) classifies the measurements as correct or incorrect. There are two incorrect types: non-existent objects estimated to exist (fale positive, FP) and existent objects not measured (false negative, FN). A FP measurement is generally caused by noisy estimations, occlusion, and ambiguous matchings. Although the path planning will be affected, it will not directly cause collisions. A FN measurement may lead to collisions with unsensed objects and is thus a good means to measure the *safety* level of depth estimation. The FP and FN boxplots in Fig. 2 shows that all three methods have a similar FP rate but that BM and SGBM have a higher FN rate (and are therefore less *safe*). Fig. 2(d) depicts the depth estimates for the three methods versus the ground truth. The BM/SGBM methods do not capture object edges as well as the stixel method, which can lead to more collisions during navigation. In sum, the stixel is fast to compute without compromising safety for navigation.

B. Ego-centric Perception Space

It is not sufficient to have accurate and fast depth estimation. Stixels used within a navigation system should promote scalable, effective processing and decision making for navigation. In particular, perception-space navigation is best suited to local planning, and not to global planning. This section describes the additional processes required to operate within a local planner. It includes the propagation of past measurements as well as the perception-space collision checking module. Computational costs are compared to the depth image equivalents.

*1) **Stixel Egocircle**:* A local planner requires the ability to accumulate, propagate and retain recently seen information. The perception-space equivalent to the local occupancy map is the egocylinder [22]. Due to the sparsity of the 1.5D stixel representation, it is only necessary to maintain and propagate a 1D curve segment in the egocylinder, which reduces the cylinder to a circle. This *stixel egocircle* has the computational advantages of the laser-scan egocircle [27].

Stixels are converted to polar coordinates, where ρ is the range of stixel relative to the origin of camera frame and the angle coordinate is $\theta = \text{atan}(x/z)$, for x and z the Cartesian coordinates of the stixels in the conventional camera frame (z-axis forward, x-axis to right and y-axis downward). Each stixel is mapped to a egocircle index ϕ_{cyl} using the homogeneous egocircle projection matrix K_{cyl},

$$\phi_{cyl} = K_{cyl} \begin{bmatrix} \theta \\ 1 \end{bmatrix} \qquad \text{where} \qquad K_{cyl} = \begin{bmatrix} f_h & | & h_c \end{bmatrix}, \quad (1)$$

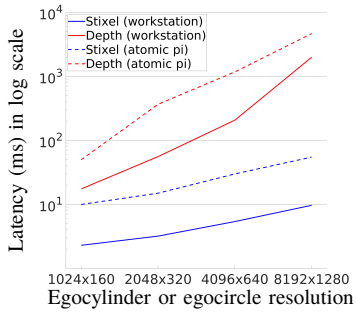

Fig. 3: Egocylindrical/egocircle propagation latency.

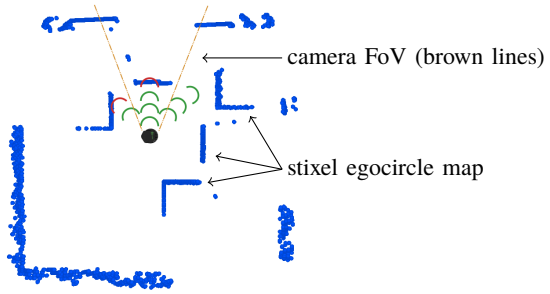

Fig. 4: Stixel-space collision checking showing valid robot trajectory poses (green) and colliding poses (red).

with $f_h = \cot(2\pi/n_{cols})$, $h_c = n_{cols}/2$, and n_{cols} predefined stixel egocircle resolution constants. Quantizing the angular index $\lfloor \phi_{cyl} \rfloor$ gives the egocircle polar measurement bin.

Stixel egocircle propagation shares similar processing to that of depth imagery with some minor modifications. Define the stixel egocircle coordinate vector $p_{cyl} = (\rho, \phi, 0)^T$ and Cartesian coordinate vector $p = (x, 0, z)^T$. In the viewer/camera frame, transformations from egocircle coordinates to Cartesian coordinates T_{e2c} and vice-versa T_{c2e} are

$$p = T_{e2c}(p_{cyl}) = \begin{bmatrix} \rho \sin(\phi) \\ 0 \\ \rho \cos(\phi) \end{bmatrix} \quad \text{and} \quad p_{cyl} = T_{c2e}(p) = \begin{bmatrix} \sqrt{x^2 + z^2} \\ \text{atan}(x/z) \\ 0 \end{bmatrix}. \tag{2}$$

Robot motion induces the camera displacement g_{move}, so that each stixel propagates to p'_{cyl} from p_{cyl} according to:

$$p'_{cyl} = T_{c2e} \circ g_{move} \circ T_{e2c}(p_{cyl}). \tag{3}$$

The new bin for the propagated point is again based on the quantization $\lfloor \phi \rfloor$. The stixel egocircle is like a 360° laserscan.

Since the stixel egocylinder maps to an egocircle based on stixels having fixed y coordinates, propogating information is cheaper that for the full ego-cylindrical image. Fig. 3 quantifies the propogation time cost for the *stixel egocircle* versus a dense depth *egocylinder* for the workstation and embedded processors. The reduction of area calculations to linear ones gives better scaling as a function of resolution, and can lead to an order of magnitude or more improvement.

*2) **Egocircle Collision Checking**:* The Planning in Perception Space (PiPS) approach [3], [22] projects a hallucinated robot into the depth image or egocylinder to perform collision checking in the image space. The stixel PiPS equivalent compares the projected robot's stixel footprint to the current stixel footprints in image-space [28] for instantaneous collision checking in the absence of memory. However, integration with the local planning *stixel egocircle*, which does have memory of past measurements, performs collision checking in 2D polar coordinates as in [27]. Figures 4 depicts the process. For candidate paths, the robot's hallucinated geometry (shown by curved green segments) is projected into the stixel egocircle space (sensed obstacles are blue). Robot stixel polar values indicating a range larger than the obstacle polar values are classified as colliding and the associated robot pose is deemed *unsafe* or inpermissible. The robot will collide if it were to follow the candidate path. Importantly, the collision checking policy applies to different robot geometries. In the case of stixels, the robot's volume is projected down to a planar footprint that gets mapped to the egocircle during collision checking. The egocircle partially resolves limited FoV issues by permitting collision checking in static worlds if the non-FoV region was recently sensed.

For images, robot volumes project to areas in the standard depth image/ego-cylinder approach so that collision checking scales with area. For a fixed vertical resolution, the ego-cylinder will scale linearly with the angular resolution. Since there is no vertical coordinate, the stixel ego-circle scales linearly. Fig. 5 plots the collision checking time for one pose versus different stixel egocircle or depth egocylinder resolutions on the workstation and Atomic Pi. The boxplots indicate that stixel egocircle collision checking scales linearly and operates faster than depth ego-cylinder collision checking.

IV. Global Path Follower Hierachical Navigation

The publicly available *move_base* package in ROS/Gazebo provides good baseline implementations for global and local planning. However, it maps the outcomes to instantaneous velocity commands, binding the fastest rate to the local planning rate. A better hierarchy has one additional time-scale that tracks local trajectories and operates faster than the local planning rate. The trajectory tracking process can exploit the low latency of PiPS collision checking to operate at a faster rate. This section describes the global path follower (GPF-X) navigation system with the additional scale level; blue blocks in Fig. 1(b). It consists of two main modules, the first of which consists of the trajectory synthesis, trajectory tracking, and state manager blocks, and corresponds to the GPF abbreviation. The second consists of the local planner block, which can flexibly be any perception-space compatible local planning system. The X is a placeholder for the local planner, two of which are described here.

A. System Overview

At the lowest *trajectory tracking* level, feedback control is applied to track the current synthesized trajectory, and the remaining future trajectory is collision checked in perception-space to confirm future safety. If the trajectory is unsafe, then a signal gets sent to the *state manager* indicating as much. Likewise a signal is sent when the trajectory segment

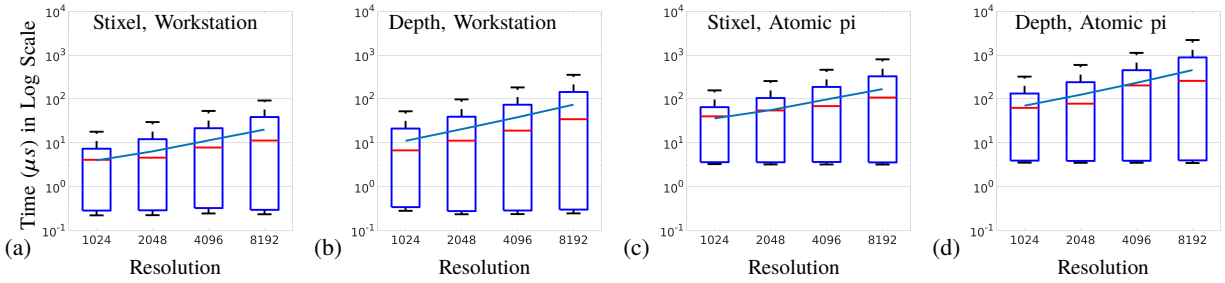

Fig. 5: Timing for each collision checking with stixel egocircle and depth egocylinder on the workstation and Atomic Pi. The log-scale x-axis is the angular resolution of the egocircle or egocylinder. The log-scale y-axis units of μs.

has been almost fully tracked. The *state manager* handles global and local re-planning plus manages the switched state of the system (*global following* versus *local synthesis*). State update consequences are transmitted to the global and local planners.

The *global planner* block operates much like the standard global planners. It finds a global path based on the current map, which gets updated based on the stixel measurements. When triggered, new global paths are recomputed based on the provided robot and specified goal poses. Skipping the *local planner* details for now, the *trajectory synthesis* takes a candidate path and synthesizes a dynamically-feasible trajectory from the path if it is not a fully specified pose and control trajectory. Synthesis involves simulating a trajectory tracking feedback controller along the candidate path. If the trajectory requires violating velocity or control constraints, then the *state manager* will be informed to trigger *local synthesis*.

When the system is *global following*, the local planner passes along a short segment of the global path for *trajectory synthesis*. Prior to doing so, it checks whether the global path sends the robot backwards. If so, it will trigger the *state manager* to switch to *local synthesis* mode. This is to avoid reversing into potentially unknown obstacles. When the state is *local synthesis*, the *local planner* will engage a short-term, local planner at the scale of the egocircle sensing radius. This process continues until the robot converges onto the current global path and triggers a state switch to *global following*.

B. Local Planner

This section sketches two perception-space GPF compatible planners inspired from [3], [27]. They process the *stixel ego-circle* to identify a candidate direction of travel, from which is synthesized a local trajectory. Both rely on the detection of navegable gaps per [27].

1) PiPS Ray: The PiPS ray (PR) local planner modifies [3] to operate as a local planner. Rather than contemplate a fixed set of forward paths, it uses the detected gaps to establish a preferred direction of travel towards the most promising gap. A local goal is established at the gap and a fixed set of single-segment Dubins paths directed towards the local goal are synthesized The paths are scored and collision-checked, with the best collision-free trajectory chosen. Fig. 6(a) depicts a single snapshot of the PiPS ray selection process.

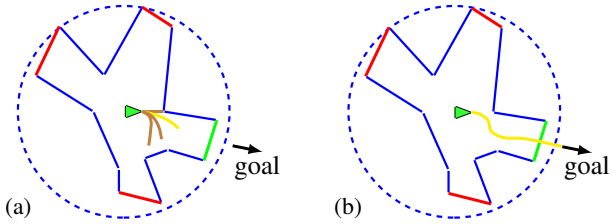

Fig. 6: Perception-space local planner visualization (left: PiPS ray, right: Potential-Gap). The black arrow is the local goal. Red curves and green curves are gaps found, with green indicating the selected gap. The final local trajectory is yellow.

2) Potential-Gap: The potential gap (PG) local planner selects the one gap that optimizes a hypothesized cost-to-go, then builds a local potential whose vector field attracts the robot to the local goal and repulses it from nearby egocircle obstacle points. The gradient assumes a holonomic robot, but the tracking controller is nonholonomic. The small spatial scale of the problem prevents some of the issues of potential methods from arising. Fig. 6(b) depicts a single snapshot of the Potential-Gap trajectory synthesis process.

V. Experiments

A. Environment and Benchmark Setups

Monte Carlo runs test the performance stixels versus block matching, as well as stixels with different planners, on a workstation and an embedded device in both cases. The tests are performed in ROS/Gazego simulation environments. There are 4 different scenarios to accomplish benchmarks. Three of them are the sector, campus and office worlds from [22] with good surface features. We also have an additional scenario (dense) that spawns many obstacles in a 20x20 empty world. The minimum distance between two obstacles is a parameter controlling the obstacle density of the environment. An additional variable testing the algorithms' ability to navigate with different robot shapes selects from two different platforms, a cylinder-shaped Turtlebot and a differentrial-drive robot the shape of the Pioneer3 [22] called "Box Turtle." Since the geometry size of the Box Turtle is different from the Turtlebot, we enlarge the minimum obstacle distance from 1m (Turtlebot) to 1.5m (Box Turtle)

TABLE I: Success rate for **Turtlebot** on Workstation/Atomic Pi.

Method	DW	CW	OW	SW	DW	CW	OW	SW
BM Laser TEB	98	82	96	90	92	48	64	88
Stixel Laser TEB	94	90	100	98	84	56	52	90
Stixel GPF PR	100	98	100	100	92	94	98	96
Stixel GPF PG	100	100	98	100	94	98	98	100

TABLE II: **Box Turtle** on workstation/Atomic Pi.

DW	CW	OW	SW	DW	CW	OW	SW
100	92	90	92	100	50	54	86
100	92	94	92	100	62	50	84
100	96	96	92	100	94	94	88
100	98	100	98	100	96	96	88

in the experiments. The maximum number of obstacles for *dense* world is 500 and for the rest is 50.

The benchmarks are performed with four different navigation approaches, BM laser TEB, stixel laser TEB, stixel GPF-PR and stixel GPF-PG. TEB stands for ROS move_base hiearchical navigation with the TEB local planner. TEB accepts laserscan as the sensor input, which is easily generated from stereo BM and stixel estimation depth measurements. Each scenario and navigation method is run 50 times. The success rate is computed from these runs. The same benchmark is applied for workstation and Atomic Pi. SGBM is not used since it does not run in real-time on the Atomic Pi. In order to guarantee the safety with the maximum sensor rate, all methods are bound to the 7Hz stixel compute rate on the Atomic Pi. For TEB, this rate applies to the local module. For GPF-X, it applies to the following module. For even comparison, we also use 7Hz on the workstation though some of the methods can run much faster, especially GPF-X.

B. Benchmark Results and Discussions

Tables I & II give the success rate of Turtlebot and Box Turtle with different navigation approaches on different hardware. The stixel GPF-X methods have high success rates across the board, whereas move_base TEB cannot perform well on the Atomic Pi with different sensor inputs, especially in the campus and office scenarios where the paths to navigate are longer. The table provides evidence in support of the paper's hypothesis. To measure this more concretely, Fig. 7 quantifies the performance gap between the workstation results and the Atomic Pi results for the different worlds and robot platforms. The mean values across the navigation scenarios for the workstation are also noted. The GPF-X methods have a change in success rate of less than 5%, while TEB is around 20%. The horizontal lines in the figure are the 95% confidence intervals for the aggregate TEB and GPF-X scores. GPF-X is more consistent across the two compute devices.

The high success rate for Box Turtle indicates that stixel GPF-X can handle non-circular robot geometries. Looking at TEB for the workstation, stixel laser TEB has a higher average performance than BM laser TEB. It suggests translation of the improved safety of stixel estimation to navigation.

1) Timing: To test the earlier assertion that GPF-X better exploits perception space timing, Fig. 8 shows the distribution of TEB and GPF-X with stixel measurements on the Atomic Pi. TEB is bound to the local planner rate, which has a high latency (120ms mean) and high variance. In contrast, GPF-X is bimodal with the majority of the cases being spent in the low latency, perception-space path confirmation mode (30ms mean), which performs efficient collision checking

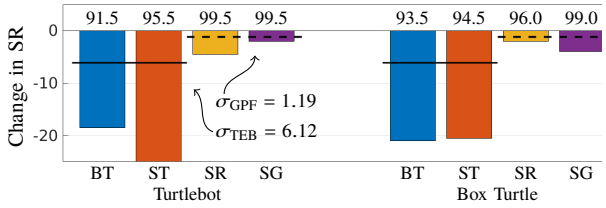

Fig. 7: Performance gap between Workstation and Atomic Pi.

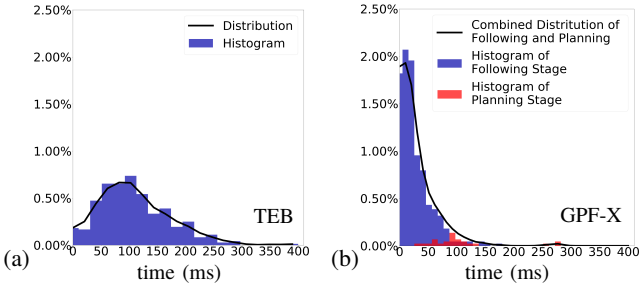

Fig. 8: Time distribution of navigation for stixel TEB and GPF-X on the atomix pi. Bars are the histograms of timing and continuous lines are the smoothed time distributions. (a) TEB (b) GPF-X with split timing histograms for planning and following stages, and combined time distribution.

along the current trajectory. The fast confirmation times enhance navigation safety, by enabling faster switching to local planning mode and global replan triggering. Only a small percent of the time is spent in local planning mode (6.5% total, in red). It has a slightly lower run-time (113ms mean) than TEB planning.

GPF-X has nice bimodal latency properties, whereas the TEB hierarchical navigation framework is unimodal and limited to how low the local planner can be in time/latency.

VI. Conclusion

Stixels have been utilized in assisting autonomous vehicle detection algorithms, but has not been explore as a free-space sensing modality. This paper extended the ego-centric PiPS representation to stixels to derive low latency stixel-based module for perception-space planning. The sparse representation of stixels permits fast, scalable feasibility checking within the perception space for different robot geometries, and fast temporal propgation of historical readings. When integrated into a hierarchical navigation system attuned to perception-space planning, stixels demonstrate more consistent navigation performance across different compute platforms. The consistency arises from the bimodal properties of properly designed perception-space local navigation modules.

References

[1] H. Badino, U. Franke, and D. Pfeiffer, "The Stixel World - A Compact Medium Level Representation of the 3D-World," in *Joint Pattern Recognition Symposium*. Springer, 2009, pp. 51–60.

[2] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416 – 442, 2015.

[3] J. S. Smith and P. A.Vela, "PiPS: Planning in perception space," in *IEEE International Conference on Robotics and Automation*, May 2017, pp. 6204–6209.

[4] H. Hirschmuller, "Stereo Processing by Semiglobal Matching and Mutual Information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, Feb 2008.

[5] A. Tonioni, F. Tosi, M. Poggi, S. Mattoccia, and L. D. Stefano, "Real-Time Self-Adaptive Deep Stereo," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019.

[6] S. Khamis, S. Fanello, C. Rhemann, A. Kowdle, J. Valentin, and S. Izadi, "StereoNet: Guided Hierarchical Refinement for Real-Time Edge-Aware Depth Prediction," in *European Conference on Computer Vision*, September 2018.

[7] S. Jin, J. Cho, X. D. Pham, K. M. Lee, S. Park, M. Kim, and J. W. Jeon, "FPGA Design and Implementation of a Real-Time Stereo Vision System," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 1, pp. 15–26, Jan 2010.

[8] Y. Li, C. Yang, W. Zhong, Z. Li, and S. Chen, "High throughput hardware architecture for accurate semi-global matching," in *Asia and South Pacific Design Automation Conference*, Jan 2017, pp. 641–646.

[9] D. Hernandez-Juarez, A. Espinosa, J. C. Moure, D. VÃązquez, and A. M. LÃşpez, "GPU-Accelerated Real-Time Stixel Computation," in *IEEE Winter Conference on Applications of Computer Vision*, 2017, pp. 1054–1062.

[10] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, "Fast stixel computation for fast pedestrian detection," in *European Conference on Computer Vision*. Springer, 2012, pp. 11–20.

[11] M. Enzweiler, M. Hummel, D. Pfeiffer, and U. Franke, "Efficient Stixel-based object recognition," in *IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 1066–1071.

[12] F. Erbs, B. Schwarz, and U. Franke, "From stixels to objects - A conditional random field based approach," in *IEEE Intelligent Vehicles Symposium*, June 2013, pp. 586–591.

[13] F. Erbs, B. Schwarz, and U. Franke, "Stixmentation - Probabilistic Stixel based Traffic Scene Labeling," in *British Machine Vision Conference*, 2012.

[14] L. Schneider, M. Cordts, T. Rehfeld, D. Pfeiffer, M. Enzweiler, U. Franke, M. Pollefeys, and S. Roth, "Semantic Stixels: Depth is not enough," in *IEEE Intelligent Vehicles Symposium*, June 2016, pp. 110–117.

[15] R. Benenson, R. Timofte, and L. Van Gool, "Stixels estimation without depth map computation," in *IEEE International Conference on Computer Vision Workshops*, Nov 2011, pp. 2010–2017.

[16] D. Murray and C. Jennings, "Stereo vision based mapping and navigation for mobile robots," in *International Conference on Robotics and Automation*, vol. 2, April 1997, pp. 1694–1699 vol.2.

[17] K. Schmid, T. Tomic, F. Ruess, H. HirschmÃijller, and M. Suppa, "Stereo vision based indoor/outdoor navigation for flying robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 3955–3962.

[18] D. Scaramuzza, M. C. Achtelik, L. Doitsidis, F. Friedrich, E. Kosmatopoulos, A. Martinelli, M. W. Achtelik, M. Chli, S. Chatzichristofis, L. Kneip, D. Gurdan, L. Heng, G. H. Lee, S. Lynen, M. Pollefeys, A. Renzaglia, R. Siegwart, J. C. Stumpf, P. Tanskanen, C. Troiani, S. Weiss, and L. Meier, "Vision-Controlled Micro Flying Robots: From System Design to Autonomous Navigation and Mapping in GPS-Denied Environments," *IEEE Robotics Automation Magazine*, vol. 21, no. 3, pp. 26–40, Sep. 2014.

[19] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics Automation Magazine*, vol. 4, no. 1, pp. 23–33, March 1997.

[20] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *IEEE International Conference on Robotics and Automation*. IEEE, 1993, pp. 802–807.

[21] C. RÃűsmann, F. Hoffmann, and T. Bertram, "Timed-Elastic-Bands for time-optimal point-to-point nonlinear model predictive control," in *European Control Conference*, July 2015, pp. 3352–3357.

[22] J. S. Smith, S. Feng, F. Lyu, and P. A. Vela, "Real-time egocentric navigation using 3d sensing," in *Machine Vision and Navigation*, O. Sergiyenko, W. Flores-Fuentes, and P. Mercorelli, Eds. Springer, 2020.

[23] L. Matthies, R. Brockers, Y. Kuwata, and S. Weiss, "Stereo vision-based obstacle avoidance for micro air vehicles using disparity space," in *IEEE International Conference on Robotics and Automation*, May 2014, pp. 3242–3249.

[24] A. J. Barry, P. R. Florence, and R. Tedrake, "High-speed autonomous obstacle avoidance with pushbroom stereo," *Journal of Field Robotics*, vol. 35, no. 1, pp. 52–68.

[25] A. T. Fragoso, C. Cigla, R. Brockers, and L. H. Matthies, "Dynamically feasible motion planning for micro air vehicles using an egocylinder," in *Field and Service Robotics*, M. Hutter and R. Siegwart, Eds. Springer International Publishing, 2018, pp. 433–447.

[26] A. T. Fragoso, "Egospace Motion Planning Representations for Micro Air Vehicles," Ph.D. dissertation, California Institute of Technology, Pasadena, CA, 10 2018.

[27] J. S. Smith, R. Xu, and P. Vela, "egoTEB: Egocentric, Perception Space Navigation Using Timed-Elastic-Bands," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 2703–2709.

[28] J. H. Hwang, "Vision-based autonomous navigation in medium level representation," Master's thesis, Georgia Institute of Technology, 777 Atlantic Drive NW, Atlanta, GA 30332, 12 2017.