2017 13th IEEE Conference on Automation Science and Engineering (CASE)
Xi'an, China, August 20-23, 2017

Obstacle Detection and Avoidance for Autonomous Bicycles

Mingguo Zhao[1] , Sotirios Stasinopoulos[2] and Yongchao Yu[3]

Abstract— **Obstacle detection and avoidance is an essential part in the field of autonomous vehicles. In this paper, we propose a novel methodology for obstacle detection and avoidance for the rapidly-developing category of autonomous bicycles. This comprises the bicycle-specific design of our Light Detection And Ranging(LiDAR)-based sensor system and our obstacle detection technique, including the segmentation of the surrounding environment 3D point cloud into discrete obstacle clusters within the bicycle's path and a method for detecting areas containing liquids on the road surface. Moreover, our obstacle avoidance technique is introduced, consisting of a local path planning and a path following part, enabling our bicycle to avoid the closest obstacle lying on the bicycle's user-defined route. The proposed method takes into account the unique dynamic characteristics of bicycles, and the results of the simulation and outdoor experiment confirm its efficacy and promising behavior.**

Index Terms— **Autonomous bicycle, LiDAR, Obstacle detection, Obstacle avoidance**

I. INTRODUCTION

In the last decade a lot of progress has been made in the field of autonomous vehicles, especially their evolution from the experimental level to everyday applications. In order for those vehicles to be reliable, they need to be aware of their environment, detect the surrounding obstacles inside it and bypass them if they are blocking the safe passage, while following the user-designed route. While most researchers are attempting to bring forth a reliable and user-friendly driverless car version, a great lot can be gained by the research of the automation of other vehicle types as well.

Bicycles are characterized by several special attributes that differentiate their environment perception requirements compared to cars and other vehicles. Firstly, their long, narrow and light structure prevents the addition of multiple, heavy and overly-complicated sensor systems. Moreover, given their size and design, bicycles are capable of altering their direction relatively easily and due to their usually low cruising speeds, they should be able to cruise closely to other vehicles. Additionally, given that bicycles are inherently unstable, their balance is greatly important. This makes detecting potentially dangerous areas on the road surface more crucial for a smooth cruise compared to four-wheeled vehicles, increasing the detection problem difficulty. For that reason, road areas with liquids and especially ones with water on the road surface, such as water puddles that are very common on city streets, need to be recognized with accuracy and circumvented, so that bicycles do not slip or skid while crossing them.

Therefore, the automation of the entire process of obstacle detection and avoidance, along with its navigation and path planning is a virtually unexplored research area for autonomous bicycles, especially since bicycles as smart means of transportation have only recently come to the forefront of the automation field. This paper is part of a research articles series that examines autonomous bicycles, their control, obstacle detection and avoidance and path planning systems. Firstly, we introduce our prototype's sensor system design, our methodology for the detection of surrounding objects, including the recognition of three-dimensional obstacles within the bicycle's safe passage, along with a novel technique for detecting areas of liquid on the road surface. In addition, we propose our new obstacle avoidance technique that contains local path planning and path following. Finally, we provide our experimental results that validate our autonomous bicycle system's reliable function.

II. RELATED WORK

Regarding the detection of obstacles within a moving vehicle's environment, several research efforts have been made; however, only during the last two decades significant progress has been achieved. This is mainly due to the recent introduction of highly accurate laser sensors and depth cameras and the creation of 3D representations of the surrounding environment with the use of point clouds. Several early attempts assembled 3D point clouds by rotating LRS and applied simple 2D occupancy grids [1]. Later, more elaborate classification methods were used on the point clouds, Support Vector Machines (SVMs) [2], or point descriptor extraction techniques [3]. Some teams focused on the isolation of the road surface plane in order to examine which elements on it should be considered as obstacles [4]. For the extraction of the road surface, most efforts make use of the RANdom SAmple Consensus (RANSAC), introduced by Fischler [5]. This algorithm has been used in numerous cases, such as for wall detection and road lane tracking [6]. Recently, apart from LRSs, the use of RGB-D cameras has also become more popular, but mainly for indoor problems due to the limitations of that technology that produces poor results under bright sunlight. Despite previous work by our team that mainly analyzed 2D obstacle detection and avoidance [7], [8], further research into 3D and road surface obstacle recognition for autonomous bicycles and akin vehicles is needed, which will be examined in this paper.

[1]Mingguo Zhao is with the Department of Automation, Tsinghua University, Beijing, 100084 China mgzhao@mail.tsinghua.edu.cn
[2]Sotirios Stasinopoulos is with the Department of Automation, Tsinghua University, Beijing, 100084 China stasinopouloss10@mails.tsinghua.edu.cn
[3]Yongchao Yu is with the Department of Automation, Tsinghua University, Beijing, 100084 China yuyc15@mails.tsinghua.edu.cn

For obstacle avoidance in unknown environments, many methods have been proposed. Among them, one of the most common approaches is the artificial potential field [9]. This method constructs a potential field where the goal point provides attractive force and the obstacles provide repulsive forces. The resultant forces drive the mobile robot to the goal and avoid the obstacles at the same time. The method has a simple theory and a high real-time performance, but some problems also exist, such as local minimum ambiguity, unreachable goal, and so on. Many improvements were attempted to solve these problems [10], [11]. In addition, model predictive control [12] is also an effective method, which considers the kinematics or even dynamics of the mobile robots and turns the problem into an optimization problem. The results are good, but the calculation is more complicated. Other approaches, such as sliding mode control [13] and "follow the gap" method [14], have also been used, but none of them has been specifically designed for autonomous bicycles, as has been accomplished in this paper.

The rest of this paper is organized as follows; the sensor integration and the environment obstacle detection modeling are described by Section III. The obstacle avoidance method and its simulation results are shown in Section IV. Our experimental setup and results are presented in Section V. Finally, Section VI offers conclusions for our system's efficient behavior and a future work discussion.

III. SENSOR SYSTEM & DETECTION METHODOLOGY

Our system's overall design, focusing on the sensor integration, rotational capability and functional efficiency, as well as the methodology describing its ability to detect obstacles, required a thorough study and implementation in order to befit the autonomous bicycles' specifications.

A. Sensor System

Numerous modern autonomous vehicle design schemes integrate various environment monitoring sensors and afterwards combine the different measurement types using models for sensor fusion. In contrast, we select a minimal sensor system design and incorporate a single outdoor SICK LMS-111 LiDAR of 270° Field of View (FoV). That is because, based on bicycles' long, narrow and light structure, the integration of numerous, heavy sensors would be harmful to the aforementioned advantageous characteristics. Moreover, by adhering to one essential LiDAR, the autonomous bicycle can remain a cost-effective, affordable ground vehicle. Finally, since a high detection speed of dynamic obstacles is a necessity, by avoiding time-consuming sensor fusion, we are able to perform in real-time. Of course, the use of a single LRS has disadvantages as well, since the sensor could make detection errors, but based on modern sensor accuracy, those errors could only be of too short duration to adversely affect our system's performance.

As seen in Fig. 1(a), the LRS, its rotating frame and the corresponding motor are adjusted on the bicycle's front, in the place of the headlight, to monitor both the frontal and lateral areas simultaneously, covering the largest possible FoV without hindering the steering or being obstructed by a rider. We design our sensor to rotate vertically while providing horizontal scans of the environment. Applying a more complex rotation pattern is infeasible in the limited space on the body of the bicycle and would interfere with the self-balance and steering control; therefore we avoid it. To monitor the exact 3D rotation of our LRS, we attach an XSens Attitude and Heading Reference System (AHRS) below our sensor.

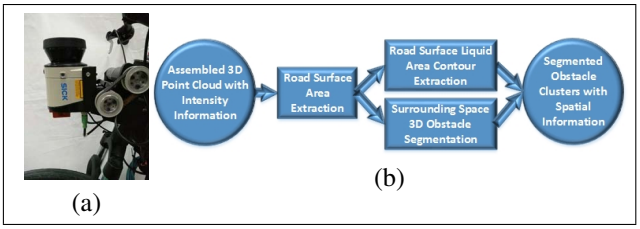

Fig. 1. (a) LiDAR mounting and rotation system design and (b) obstacle detection scheme.

B. Detection Model

In our three-dimensional obstacle detection model, we firstly create a 3D representation of the surrounding environment, by rotating our on-board 1D LRS, collecting all the consecutive scans and combining them into a single Point Cloud. Afterwards, we continue to segment the Point Cloud into separate regions, with the first region to be isolated being the road surface, using the SAmple Concensus (SAC) methodology, as we will explain in the following sections. The presence of liquid clusters is detected on the retrieved ground surface point cloud. Finally, we perform further segmentation of the remaining surrounding environment objects into clusters according to their spatial definitions, in order to complete the obstacle detection and provide the bounding boxes of the areas the bicycle should avoid during navigation.

1) 3D Point Cloud Reconstruction: Although in recent years, 3D LiDAR sensors are becoming more popular, the cost of using such sensors still remains unapproachable for everyday commercial use. We choose a much cheaper 1D LiDAR in order to keep the overall cost of the autonomous bicycle system low. The use of a 3D LRS is still possible though, with the remaining parts of our methodology remaining the same. Our SICK LiDAR has a scanning frequency of $f_{scan} = 50Hz$ and we choose to have a period of $T_{rot} = 2sec$ for a full up-down rotation, in order to get a relatively good scan of the environment objects' structure. We combine the separate scans to create a Point Cloud representation of the environment according to the pitch angle of the sensor θ and the overall bicycle roll angle φ. The measurement of θ and φ is provided by our AHRS, and transformation data about the LRS's rotation is provided at each instant. The scans are combined using the Robot Operating System (ROS), resulting in a Point Cloud of 54000 points.

After having managed to assemble our 3D Point Cloud that also includes intensity/reflectivity information for each point,

we include a stage of preprocessing to remove unnecessary regions of the Point Cloud and reduce its size for faster process. Thus, we create a Pass-through Filter to discard the areas of the Point Cloud that are of no interest for obstacle detection purposes. Given the range of our LRS being $20m$, we need to limit the data to be processed within a bounding box of $[x_{bound} \times y_{bound} \times z_{bound}]$ around the bicycle. In most of our experiments we use a $[20m \times 4m \times 1m]$ because the most significant areas for obstacle detection reside in the front of the bicycle, within its path and not higher than the height of the bicycle.

2) Road Surface Liquid Detection: As explained earlier, potentially dangerous regions of a detected road surface need to be further classified, such as areas with liquids, especially water. For vehicles like bicycles and akin inherently unstable two-wheel vehicles, these regions should be avoided like obstacles. In our work, we propose the utilization of the Point Cloud data provided by our LRS, namely the Point 3D location information (XYZ) and the Point Reflectivity/Intensity information (I), to detect such areas. Following our preprocessing, we proceed to extract the road surface area, using the SAC method to isolate points that belong to a plane, which is similar to RANSAC [5], in the way that incoming points are classified as inliers or outliers of a model, but this time the model is not random, but that of a plane instead. As a classification criterion we use the Surface Normals of the points, which are calculated within each point's neighborhood, and thus extract the plane inliers.

In order to detect liquids on the road surface point cloud, we mainly use the Point Intensity and the Point Density information. After observation, the Point Intensity around a puddle of water differs greatly compared to its surroundings, which can be used to detect the contour of the puddle. Moreover, we define the density of a Point Cloud neighborhood around a point p as the total number n of neighbors p_i of point p within the neighborhood p_n of radius r_D around p. Due to the LiDAR sensor structure however, both rotating 1D LRS, like the SICK LiDAR, and 3D LRS, like the Velodyne LiDAR, produce point clouds that have more points in closer ranges and fewer points in further ranges. Therefore, in an (r, ω) polar system, where r is the range and ω the bearing, differences in point density over the r-direction are occurring naturally and their significance is smaller than differences in the direction perpendicular to r.

We suggest the usage of the Point Bilateral Density Difference (BDD) that can be calculated as

$$\begin{matrix} D_r = n_{r,+} - n_{r,-} \\ D_{r\perp} = n_{r\perp,+} - n_{r\perp,-} \end{matrix} \Rightarrow D_p = \sqrt{(w_r D_r)^2 + (w_{r\perp} D_{r\perp})^2} \quad (1)$$

where D_r, $D_{r\perp}$ are the partial Point BDDs in the direction of range r and the perpendicular direction $r\perp$, respectively, $n_{\cdot,+}$ and $n_{\cdot,-}$ are the numbers of neighboring points left and right, respectively, of the point in that specific direction. $w_\cdot$ is the weight for the partial Point BDD in the respective direction. As explained before, w_r must be much smaller than $w_{r\perp}$.

Since areas of water puddles do not contain any actual point cloud data, because layers of water thicker than $1mm$ do not reflect the light emitted by our LRS, we can use the Point BDD to detect the contours of water puddles. Therefore, through a combination of the surface point cloud's Point Intensity and Point BDD, we can detect the water puddles on the surface by setting thresholds about these two metrics and examining the clusters of points that meet these requirements.

3) 3D Obstacle Detection: After removing the road surface, the remaining points belong only to obstacles surrounding the bicycle, which need to be segmented even further into separate areas. We achieve this by using a k-dimensional tree (k-d tree) representation of the Point Cloud. k-d trees are space-partitioning data structures, useful for efficient searches [15]. Thus, we perform a search upon the Point Cloud and find areas with points that are in close spatial (Euclidean) proximity. Therefore, starting from random points as the cluster cores, we separate points according to the Euclidean distance from their neighbors, i.e. applying Euclidean clustering, and finally attain the separate obstacles.

The results of our obstacle detection technique can be seen comprehensively in Fig. 2, where a puddle of water and a human are detected as obstacles in our bicycle's path and are marked with different colors in the visualization provided by ROS's Rviz. After segmenting the obstacle clusters, we transfer information about their range, bearing, width and height (for 3D obstacles) to the obstacle avoidance part.

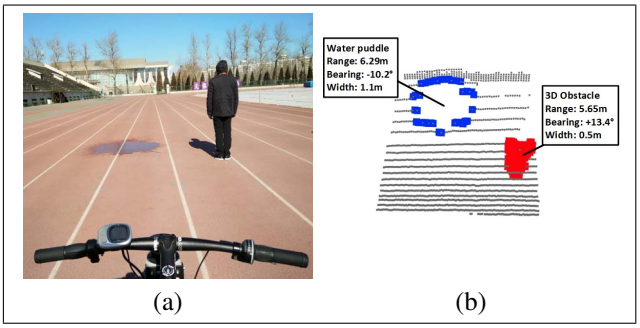

Fig. 2. (a) Original image of the obstacles and (b) Rviz result of detected obstacle clusters upon the segmented road surface point cloud, with information about the obstacle clusters annotated.

IV. PATH PLANNING & OBSTACLE AVOIDANCE

In this paper, a new method is designed for autonomous bicycles to avoid detected obstacles. This method includes two parts, namely the local path planning part and the path following part. In the case of absence of obstacles, the bicycle will follow the planned global reference path to its destination. When obstacles are detected, a local path to avoid them is planned and the bicycle will follow the new local path circumventing the obstacles and get back to the original global path afterwards. In our proposed methodology, we only consider the case of static environment obstacles.

In addition, the following assumptions are made:

(1) Every obstacle has a circular shape, so its radius r_{obs} is the half of its width w_{obs} detected by the LiDAR.

(2) Since the LiDAR is mounted in the front of the bicycle and the GPS above the rear wheel frame, the relationship

between the LiDAR's position (x_L, y_L) and the position (x_G, y_G) measured by the GPS is as follows (see Fig. 2(a)):

$$\begin{aligned} x_L &= x_G + l_{bike} \cdot \cos(\psi) \\ y_L &= y_G + l_{bike} \cdot \sin(\psi) \end{aligned} \tag{2}$$

where ψ is the bicycle's current yaw angle. In the remainder of this paper, the LiDAR's position is equivalent to the bicycle's position.

(3) The detection algorithm in Section III obtains the obstacle's width w_{obs}, bearing-local yaw angle ψ_{obs} and range-distance to the bicycle d_{obs}, thus, the global coordinates of the detected obstacle can be calculated based on this information (see Fig. 2(b)):

$$\begin{aligned} x_{obs} &= x_L + d_{obs} \cdot \cos(\psi_{obs} - \psi) \\ y_{obs} &= y_L - d_{obs} \cdot \sin(\psi_{obs} - \psi) \end{aligned} \tag{3}$$

(4) The width of the bicycle should also be taken into account. In order to consider the robot as a point to simplify the problem, the width of the bicycle is added to the obstacle radius, as illustrated in Fig. 2(b) [14]. Therefore, the new radius of the obstacle r'_{obs} becomes:

$$r'_{obs} = r_{obs} + \frac{1}{2} w_{bike} + r_{th} \tag{4}$$

where r_{th} is a constant coefficient, representing an additional safety distance, in order to ensure the bicycle does not approach the obstacles excessively.

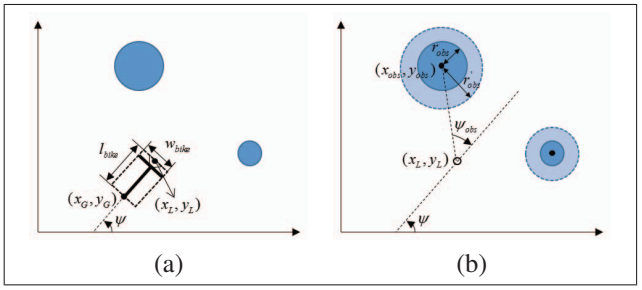

Fig. 3. (a) The original obstacles and the bicycle (b) The adjusted obstacles and bicycle models.

A. Local Path Planning

When an obstacle hinders the bicycle's passage, the most intuitive way to avoid it is by circumventing it. We decompose the whole process into four phases. In every phase, an easy to follow linear path is planned taking into account the dynamic characteristics of the bicycle, including 5 key variables, namely the coordinates of the starting point $(x_{start\_i}, y_{start\_i})$, and the terminal point (x_{term_i}, y_{term_i}), and finally, the reference yaw angle ψ_{r_i}, $i = 0, 1, 2, 3$. Fig. 4 presents our entire strategy of local path planning.

1) PHASE 0: In Phase 0, no obstacles are detected by the LiDAR, so the reference path is the predefined global linear path, and the starting and terminal points are the global starting and terminal points. The reference yaw angle is:

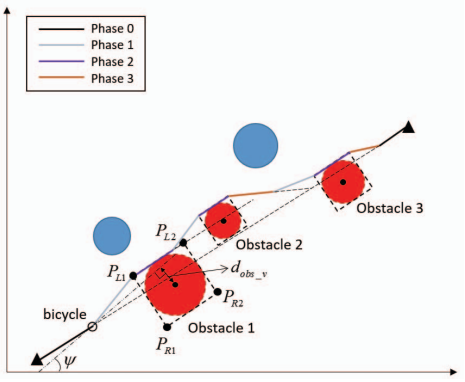

Fig. 4. Local path planning strategy

$$\psi_{r\_0} = \arcsin\left(\frac{y_{term\_0} - y_{start\_0}}{\sqrt{(y_{term\_0} - y_{start\_0})^2 + (x_{term\_0} - x_{start\_0})^2}}\right) \tag{5}$$

When the LiDAR detects obstacles and some of them coincide with the global path, the algorithm goes into Phase 1 (see Obstacle 1 in Fig. 4). To judge whether an obstacle coincides with the reference path, we can calculate the vertical distance $d_{obs\&path}$ between the obstacle and the linear path. If $d_{obs\&path} \leq r'_{obs}$, the obstacle coincides with the path.

2) PHASE 1: The obstacles coinciding with the current reference path are named 'dangerous obstacles'. Among them, the closest one to the bicycle is called 'the most dangerous obstacle'. Fig. 5 visually represents different kinds of obstacles. After finding 'the most dangerous obstacle', if the distance between it and the bicycle is shorter than the safe distance, the avoidance path is designed, and the bicycle will begin to avoid the obstacle. The starting point P_{start_1} of the path is the bicycle's current position (x_L, y_L). Since the shape of the obstacle is assumed to be circular, the four vertices of its circumscribing square are used as the terminal point of the path, ensuring the bicycle can avoid the obstacle.

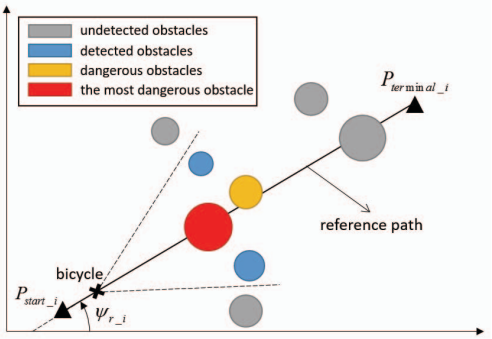

Fig. 5. Different kinds of obstacles

In Fig. 4, we can see the four vertices of the obstacle $P_{L1}, P_{L2}, P_{R1}, P_{R2}$. $P_{L1}P_{L2}$ is parallel to the global reference path. In Phase 1, the terminal point P_{term_1} is either P_{L1} or P_{R1} depending on whether the avoidance direction is left or right. There are two factors that need to be considered to decide the avoidance direction, which are the side of the

bicycle on which the obstacle lies and the roll direction of the bicycle at the given time instant. As illustrated in Fig. 4, the lateral distance between the obstacle and the bicycle is:

$$d_{obs\_v} = \frac{\tan(\psi)\cdot x_{obs} - y_{obs} + y_L - \tan(\psi)\cdot x_L}{\sqrt{1+\tan(\psi)^2}} \tag{6}$$

If $d_{obs\_v} > k_n \cdot r'_{obs}$, the avoidance direction is left, whereas if $d_{obs\_v} < -k_n \cdot r'_{obs}$, it is right. k_n is a constant coefficient and $0 \le k_n \le 1$. In these two cases the side on which the obstacle lies is more important. If $|d_{obs\_v}| < k_n \cdot r'_{obs}$, the bicycle's roll direction decides the avoidance direction. If the bicycle is tilted to the left, then we let it turn left to avoid the obstacle and vice versa. This approach can avoid large steering and make the bicycle more stable. The reference yaw angle is calculated as follows:

$$\psi_{r\_1} = \arctan\left(\frac{y_{term\_1} - y_L}{x_{term\_1} - x_L}\right) + \psi_{r\_0} \tag{7}$$

When the bicycle approaches P_{term_1}, the algorithm goes into Phase 2.

3) PHASE 2: In phase 2, the bicycle will drive along one side of the 'the most dangerous obstacle' found in Phase 1. Therefore, the starting point P_{start_2} of the planned path is the terminal point P_{term_1} of Phase 1. The terminal point P_{term_2} is either P_{L2} or P_{R2} depending on the avoidance direction in Phase 1. The reference yaw angle $\psi_{r\_2} = \psi_{r\_0}$.

If the LiDAR detects other obstacles and there are 'dangerous obstacles' among them during this phase (see Obstacle 2 in Fig. 4), the algorithm goes back to Phase 1 to avoid the new 'the most dangerous obstacle'. If the bicycle approaches P_{term_2} without detecting any 'dangerous obstacles', the algorithm goes into Phase 3.

4) PHASE 3: After Phase 2, the bicycle has completely avoided 'the most dangerous obstacle', so it should go back to the global reference path. The starting point P_{start_3} of the path is the bicycle's current position (x_L, y_L). The reference yaw angle $\psi_{r\_3} = \psi_{r\_0} \pm \psi_a$ depends on the avoidance direction in Phase 1. ψ_a is a constant coefficient. The terminal point P_{term_3} is the intersection of the local reference path and the global reference path.

If the LiDAR detects new obstacles coinciding with the current local path or the global reference path during this phase (see Obstacle 3 in Fig. 4), the algorithm goes back to Phase 1. If the bicycle approaches P_{term_3} without detecting any 'dangerous obstacles', the algorithm goes into Phase 0, where the bicycle restarts following the global reference path.

B. PATH FOLLOWING

At each time instant, the local path planning algorithm calculates the coordinates of the starting point $(x_{start\_i}, y_{start\_i})$, the terminal point (x_{term_i}, y_{term_i}) of the current planned linear path and the current reference yaw angle ψ_{r_i}.

Fig. 6 explains the path following strategy. We provide the distance deviation Δd, yaw angle ψ and their differential values $\Delta\dot{d}$ and $\dot{\psi}$ as feedback, and thus, when following the planned linear path, the control law is as follows:

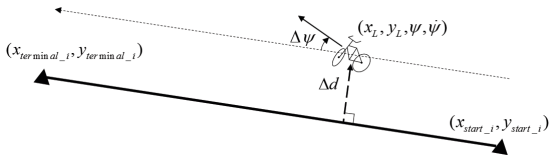

Fig. 6. Path following strategy

$$\begin{aligned}\varphi_d = {} & w_{yaw}(k_{p\_yaw}\cdot\Delta\psi + k_{d\_yaw}\cdot\dot{\psi}) \\ & + w_{dist}(k_{p\_dist}\cdot\Delta d + k_{d\_dist}\cdot\Delta\dot{d})\end{aligned} \tag{8}$$

where w_{yaw} and w_{dist} are weights of the yaw angle and distance deviation, which must satisfy: $w_{yaw} + w_{dist} = 1$.

C. SIMULATION RESULT

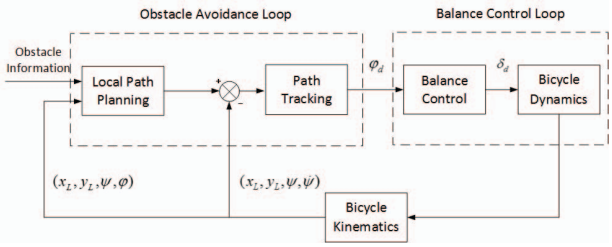

Fig. 7. Simulation system structure

Fig. 7 illustrates the structure of the entire simulation system. The bicycle dynamics and balance control algorithm can be seen in [16], and the bicycle kinematics in [17].

In the simulation, the bicycle's velocity is set to be constant, which is 2 m/s. Fig. 8 shows the simulation result, where the planned local path can avoid all the dangerous obstacles and the bicycle can follow the reference path accurately to its destination, supporting the good performance of our methodology.

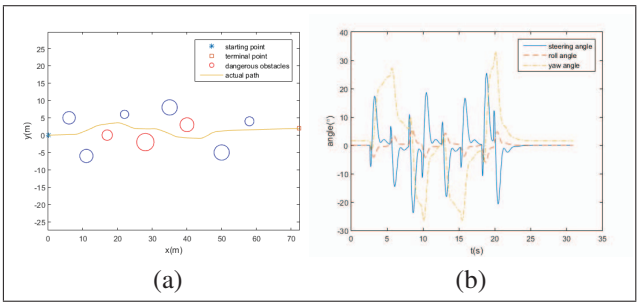

Fig. 8. (a) The actual path of the bicycle (b) The key state variables of the bicycle.

V. EXPERIMENTAL SETUP & RESULTS

Based on the autonomous bicycle prototype designed and built by our Robot Control Laboratory, seen in Fig. 9(a), our system consists of an Intel NUC for the implementation of our obstacle detection algorithm, receiving and analyzing the LiDAR scans over ROS. The motors and their corresponding actuators are controlled by a Compact RIO (cRio), which ensures the self-stabilization and the obstacle avoidance. An MTi-300 AHRS for measuring the bicycle roll rotation φ,

Fig. 9. (a) The implemented autonomous bicycle and (b) our experimental venue for obstacle detection and avoidance.

yaw rotation ψ and the LRS tilt θ, and a differential GPS for positioning are also integrated.

Given the above configuration, we test our system's advantageous behavior on a running track to provide a semi-controlled environment. Our bicycle moves in a linear course with an average speed of $2.2m/s$ and the LRS rotates with a tilt angle range of 40deg to create the 3D point cloud. For our experiment, we use a puddle of water and a human as the two kinds of obstacles to avoid, as can be seen in Fig. 9(b). The experiment results are shown in Fig. 10. The bicycle is able to detect the water puddle and the people coinciding with the reference path and avoid them successfully, which suggests that the obstacle detection and avoidance method proposed in this paper is effective.

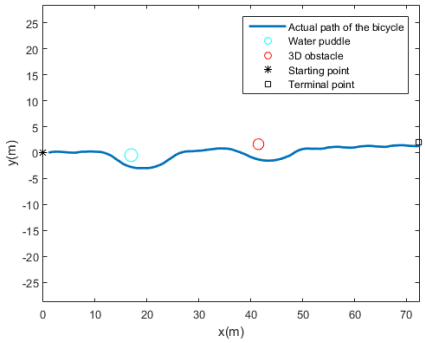

Fig. 10. Obstacle detection and avoidance experiment result

VI. CONCLUSIONS AND DISCUSSION

In this paper, we have introduced our autonomous bicycle prototype and its novel obstacle detection and avoidance methodology. More specifically, we described our sensor system, particularly designed to comply with the bicycle's requirements, and our obstacle detection technique that relies on the segmentation of the surrounding environment 3D point cloud into different obstacles on the bicycle's path along with a method for detecting areas with liquid on the road surface. Furthermore, we analyzed our obstacle avoidance methodology that comprises local path planning and path following in order to circumvent the closest obstacle on the bicycle's predefined course. Finally, we tested our autonomous bicycle system's performance on an outdoors running track and verified its reliable behavior.

Despite the promising behavior of our system, there are several ways to improve our obstacle detection and avoidance methodology. As observed during our experiments, there is a significant delay inserted by the 3D reconstruction technique, so using a 3D LRS or a faster 3D reconstruction approach could be examined. In addition, our modeling only takes into account a bicycle moving at a constant speed within an environment of static obstacles. Future research should focus on more complex obstacle avoidance techniques, tackling the varying bicycle velocity and dynamic environment problem.

REFERENCES

[1] M. Whitty, S. Cossell, K. S. Dang, J. Guivant, and J. Katupitiya, "Autonomous navigation using a real-time 3d point cloud," in *2010 Australasian Conference on Robotics and Automation*, 2010, pp. 1–3.

[2] M. Himmelsbach, T. Luettel, and H.-J. Wuensche, "Real-time object classification in 3d point clouds using point feature histograms," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 994–1000.

[3] K. Klasing, D. Wollherr, and M. Buss, "Realtime segmentation of range data using continuous nearest neighbors," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 2431–2436.

[4] H. Guan, J. Li, Y. Yu, Z. Ji, and C. Wang, "Using mobile lidar data for rapidly updating road markings," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 16, no. 5, pp. 2457–2466, 2015.

[5] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[6] A. Borkar, M. Hayes, and M. T. Smith, "Robust lane detection and tracking with ransac and kalman filter." in *ICIP*, 2009, pp. 3261–3264.

[7] S. Stasinopoulos and M. Zhao, "Laser-based obstacle avoidance and road quality detection for autonomous bicycles," in *Proceedings of the 2015 Chinese Intelligent Automation Conference*. Springer, 2015, pp. 215–223.

[8] S. Stasinopoulos, M. Zhao, and Y. Zhong, "Human behavior inspired obstacle avoidance & road surface quality detection for autonomous bicycles," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2015, pp. 2121–2126.

[9] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[10] C. Pozna, F. Troester, R. E. Precup, J. Tar, K. Zsef, and S. Preitl, "On the design of an obstacle avoiding trajectory: Method and simulation," *Mathematics & Computers in Simulation*, vol. 79, no. 7, pp. 2211–2226, 2009.

[11] L. Tang, S. Dian, G. Gu, and K. Zhou, "A novel potential field method for obstacle avoidance and path planning of mobile robot," in *IEEE International Conference on Computer Science and Information Technology*, 2010, pp. 633–637.

[12] Y. Yoon, J. Shin, H. J. Kim, Y. Park, and S. Sastry, "Model-predictive active steering and obstacle avoidance for autonomous ground vehicles," *Control Engineering Practice*, vol. 17, no. 7, pp. 741–750, 2009.

[13] A. S. Matveev, H. Teimoori, and A. V. Savkin, "A method for guidance and control of an autonomous vehicle in problems of border patrolling and obstacle avoidance," vol. 47, no. 3, pp. 515–524, 2011.

[14] V. Sezer and M. Gokasan, "A novel obstacle avoidance algorithm:follow the gap method," *Robotics and Autonomous Systems*, vol. 60, no. 9, pp. 1123–1134, 2012.

[15] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[16] J. He, M. Zhao, and S. Stasinopoulos, "Constant-velocity steering control design for unmanned bicycles," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2015, pp. 428–433.

[17] Y. Yu and M. Zhao, "An improved path following algorithm of the autonomous bicycle," *Journal of Huazhong University of Science and Technology (Natural Science Edition)*, vol. 43, sup. 1, pp. 345–350, 2015.