Senseable City Lab :.:: Massachusetts Institute of Technology

This paper might be a pre-copy-editing or a post-print author-produced .pdf of an article accepted for publication. For the definitive publisher-authenticated version, please refer directly to publishing house's archive system

Roboat II: A Novel Autonomous Surface Vessel for Urban Environments

Wei Wang, Tixiao Shan, Pietro Leoni, David Fernández-Gutiérrez, Drew Meyers, Carlo Ratti and Daniela Rus

***Abstract*— This paper presents a novel autonomous surface vessel (ASV), called Roboat II for urban transportation. Roboat II is capable of accurate simultaneous localization and mapping (SLAM), receding horizon tracking control and estimation, and path planning. Roboat II is designed to maximize the internal space for transport, and can carry payloads several times of its own weight. Moreover, it is capable of holonomic motions to facilitate transporting, docking, and inter-connectivity between boats. The proposed SLAM system receives sensor data from a 3D LiDAR, an IMU, and a GPS, and utilizes a factor graph to tackle the multi-sensor fusion problem. To cope with the complex dynamics in the water, Roboat II employs an online nonlinear model predictive controller (NMPC), where we experimentally estimated the dynamical model of the vessel in order to achieve superior performance for tracking control. The states of Roboat II are simultaneously estimated using a nonlinear moving horizon estimation (NMHE) algorithm. Experiments demonstrate that Roboat II is able to successfully perform online mapping and localization, plan its path and robustly track the planned trajectory in the confined river, implying that this autonomous vessel holds the promise on potential applications in transporting humans and goods in many of the waterways nowadays.**

I. INTRODUCTION

The increasing needs for water-based navigation in areas such as oceanic monitoring, marine resource exploiting, and hydrology surveying have all led to strong demand from commercial, scientific, and military communities for the development of innovative autonomous vessels (ASVs) [1]–[6]. ASVs also have a promising role in the future of transportation for many coastal and riverside cities such as Amsterdam and Venice, where some of the existing infrastructures like roads and bridges are always overburdened. A fleet of eco-friendly self-driving vessels could shift the transport behaviors from the roads to waterways, possibly reducing street traffic congestion in these water-related cities.

Much progress has been made on ASV autonomy in the last several decades [6], such as localization [1], [4], [5], object detection [7], path planning [5], [8], [9] and tracking control [10], [11]. However, current ASVs are usually developed for open waters [6], [11], [12] and thus cannot satisfactorily meet the autonomy requirements for applications in narrow and crowded urban water environments such as Amsterdam canals. Developing an autonomous system for vessels in urban waterways is more challenging than for traditional ASVs in open water environments.

This paper focuses on the design of localization and control problems for urban ASVs. First, to safely navigate in urban waterways, an ASV should localize itself with centimeter-level or decimeter-level accuracy. Current ASVs usually use GPS and IMU (fused by an extended Kalman filter (EKF) or unscented Kalman filter (UKF)) which typically results in a meter-level precision [6]. These GPS-IMU-based approaches can be unstable in urban waterways, where GPS signals are often severely attenuated. A reliable multi-sensor navigation system which includes GPS, compass, speed log, and a depth sensor to account for sensor failure was proposed, but it cannot guarantee high accuracy [13]. To date, there is no feasible solution for accurate urban ASV localization. Second, a number of tracking control methods such as sliding mode method [14], integrator backstepping method [10], [15] and adaptive control [16] have been proposed for ASVs. However, most of the current controllers are either verified by simulation or partly verified in open waters which do not care too much of the tracking accuracy. Moreover, many controllers use a kinematic model instead of a dynamical one for the vessels. The control performance will always decline a lot due to the highly nonlinearity of the water and the persistent disturbances in real environments.

Our recently launched Roboat project aims at developing a fleet of autonomous vessels for transportation and constructing dynamic floating infrastructure [17]–[20] (e.g., bridges and stages) in the city of Amsterdam. In our previous work [17], [18], we have designed a quarter-scale Roboat that was able to localize itself using LiDAR and track on the reference trajectory using NMPC. By contrast, this paper develops a new large-scale vessel, Roboat II, which is capable of carrying passengers. Moreover, we develop the new SLAM algorithm, NMHE state estimation algorithm, and adapt the NMPC for large-scale Roboat II. The main contributions of our work can be summarized as follows:

- Designing and building of a new vessel, Roboat II;
- NMHE state estimation for autonomous vessels;
- Simultaneous localization and mapping (SLAM) algorithm for urban vessels;
- NMPC-NMHE control strategy with full state integration for accurate tracking;
- Extensive experiments to validate the developed autonomy system in rivers.

This paper is structured as follows. Section II overviews

This work was supported by grant from the Amsterdam Institute for Advanced Metropolitan Solutions (AMS) in Netherlands.

W. Wang, T. Shan, P. Leoni, D. Gutierrez, D. Meyers and C. Ratti are with the SENSEable City Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA. {wweiwang, shant, leoni, davidfg, drewm, ratti}@mit.edu

W. Wang, T. Shan and D. Rus are with the Computer Science and Artificial Intelligence Lab (CSAIL), Massachusetts Institute of Technology, Cambridge, MA 02139 USA. {wweiwang, shant, rus}@mit.edu

the Roboat prototype. Section III describes the framework of the developed autonomy system for urban ASVs. NMPC and NMHE are described in Section IV. SLAM algorithm is presented in Section V. Experiments are presented in Section VI. Section VII concludes this paper.

II. Roboat Design

A. Hull Design

Roboat II is designed starting from two main principles: simple construction and inter-connectivity between vessels. The former principle leads to define the hull shape using single-curvature surfaces. Regarding the latter, the ability to connect multiple units in the water to build larger structures, side by side or perpendicularly, dictated the 1:2 ratio. In this way, two vessels can dock with their short sides on the long, lateral side of a third one if needed. Also, this means an even distribution of the location of the connectors, leaving as empty bays spaced along the sides for easy accommodation of latching modules, as shown in Fig. 1. Bolted to the

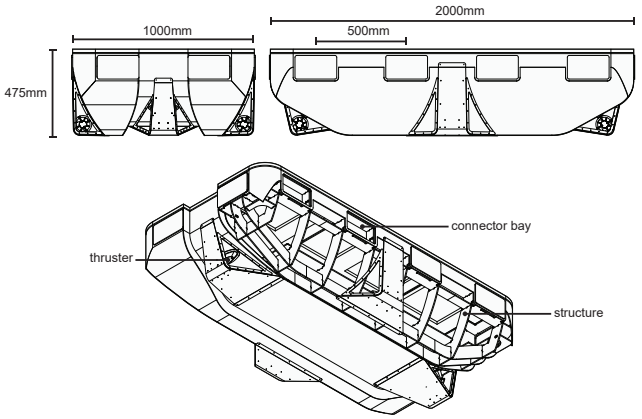

Fig. 1. Mechanical design of Roboat II.

structure, these modules are fast to replace, iteration after iteration, without further interventions on the hull itself. Being those the points where the vessel is connected, they are also particularly subjected to mechanical stress. Thus, a structural rib is placed on each of its sides to distribute the forces. With a connector every 500 mm, it is their position that dictates the internal ribs distribution.

Other than this, the need for a unit capable of maximizing the internal space and not required to move fast (cruise speed is 7.5 km/h) resulted in a bulky shape, perfectly symmetrical on the two main axes, and with the control system able to adjust the thrust to move in any desired direction. Marine plywood has been preferred because of its lower cost and easier workability. Similar to if we were using aluminum foil, the easier option for the assembly was to design the hull as a combination of CNC cut sheets, 4 to 6 mm thick and bent if needed along a single direction, rather than pressed to achieve double curvature. A fiberglass coating was still applied on the exterior, with the interior space being easily accessible through 8 waterproof hatches on the top deck. Furthermore, the whole top deck was not glued permanently to the hull but bolted instead, with a neoprene gasket in

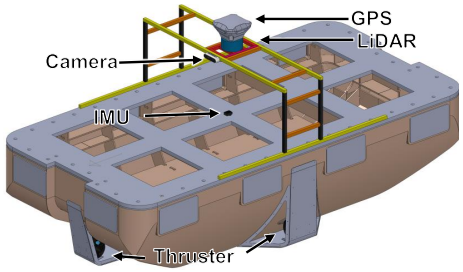

Fig. 2. Hardware design of Roboat II.

the middle. This allows accessing the internal volume in situations where the hatches are not large enough, such as for extended intervention on the inside structure or for permanent placement of large components.

B. Hardware

We use a small-form-factor barebone computer, Intel® NUC (NUC7i7DNH) as the main processor of our vessel which runs a robotics middleware, Robot Operating System (ROS). An auxiliary STM32 processor is used for converting the calculated forces from the controller to actuator signals. Roboat also has several onboard sensors, as shown in Fig. 2. More specifications of Roboat are listed in Table I.

TABLE I
Technical specifications of the prototype

Items	Characteristics
Dimension (L×W×H)	2.0 m× 1.00 m × 0.475 m
Total mass	∼ 80.0 kg
Center of gravity height	∼ 175 mm
Drive mode	Four T200 thrusters
Onboard sensors	3D LIDAR, IMU, Camera, GPS
Power supply	14.8 V, 22A·h Li-Po battery
Operation time	∼ 2.0 hours
Control mode	Autonomous/Wireless mode
Maximum speed	1.0 Body Length/s

C. Hydrodynamic Analysis with CFD Simulation

The mechanical model described in the previous sections allows us to determine the hydrostatic coefficients of the prototype. Figure 3 shows the mass displacement and wetted surface as functions of the vessel draught assuming it floats in brackish water (density ~ 1010 kg/m^3), which have a direct impact on its dynamics.

Based on the total mass and center of gravity height of the tested prototype from Table I, the expected draught of the prototype from Fig. 3 is 107 mm. Given that these Roboat units will operate in environments with wave heights below 10 cm, which yield roll and pitch moments ∼5 and ∼10 kg·m, respectively, we expect roll and pitch angles below 5°. These outcomes validate the assumption of planar motion followed in the dynamic model described in Sec. IV-A. Moreover, 3 also shows the capability to carry a much larger payload than the quarter-scale model [17]. Roboat II can carry two people on-board (see supplementary material).

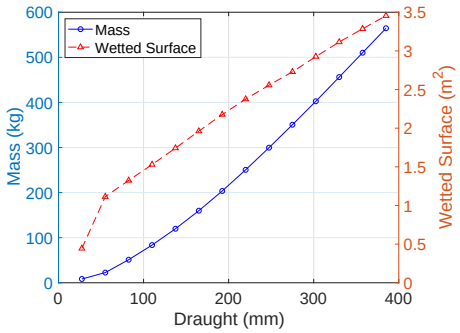

Fig. 3. Displaced mass and wetted surface values as function of draught.

Additionally, we used the SolidWorks® Flow Simulation package to evaluate numerically the hydrodynamic response of the Roboat under the design operating conditions. Figure 4 shows a side view with the pressure distribution over the hull as the Roboat moves forward at 0.8 m/s, as well as the velocity streamlines around it. The results show how the smooth hull efficiently deflects the incident flow, minimally perturbing the free surface and avoiding high dynamic pressure concentrations.

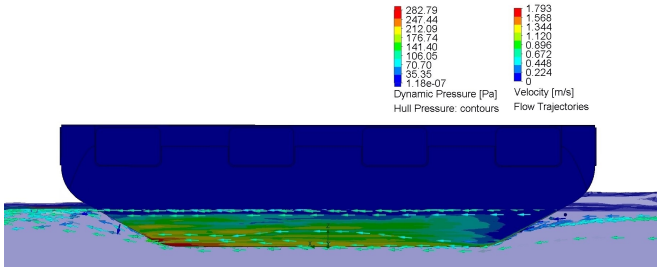

Fig. 4. CFD results from SolidWorks® Flow Simulation with the Roboat moving forward at 0.8 m/s.

III. Autonomy Framework of Urban Vessels

In this section, we describe the autonomy framework of our Roboat II which can move in urban waterways, as is shown in Fig. 5. The autonomous Roboat conducts a transportation task in the canals as follows. When a task such as passenger delivery is required from a user at a specific position, the system coordinator will assign this delivery task to an unoccupied Roboat that is closest to the passenger. When Roboat picks up the passenger, it will first plan its path to the destination and then generate a feasible path based on the current traffic condition. Then, Roboat starts to localize itself by running the developed SLAM algorithm which utilizes LiDAR, IMU, and GPS sensors. The NMPC controller will accurately and robustly track the reference trajectories from the planner during the whole task in an urban waterway. The planner will avoid obstacles if it receives obstacle information from an obstacle detector. We use a simple A* planner for the path generation and focus on SLAM, dynamics modeling and identification, NMPC tracking and NMHE state estimation in this study.

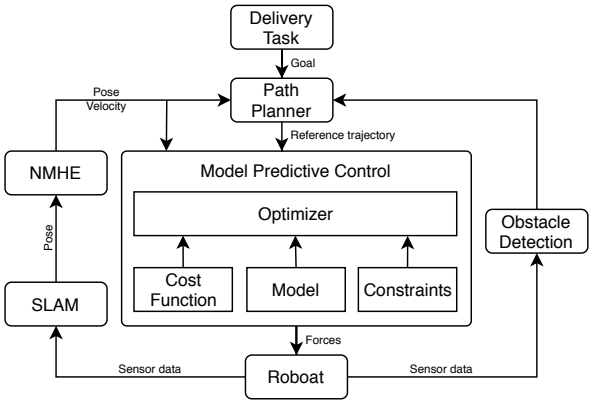

Fig. 5. Current autonomy framework of Roboat II. It mainly contains a planner,a SLAM module, an NMPC tracking module, an NMHE state estimator, and a simple object detector.

IV. Receding Horizon Control and Estimation

Nonlinear model predictive control (NMPC) is a dynamic optimization-based strategy for feedback control which determines the current control action by optimizing the system behavior over a finite window, often referred to as the prediction horizon. The NMPC controller is responsible for tracking the calculated optimal trajectory from the path planner. The states of the vessel are simultaneously estimated using a nonlinear moving horizon estimation (NMHE) algorithm. These estimated states are further fed to the NMPC, as shown in Fig. 5. The performance of the NMPC largely relies on the selected system model. Therefore, we build a dynamical model for Roboat II, and experimentally identify the unknown parameters in the model to achieve superior performance for NMPC control.

A. Dynamical Model

The dynamics of our vessel is described by the following nonlinear differential equation [17]

$$\dot{\mathbf{x}} = \mathbf{T}(\mathbf{x})\mathbf{v} \tag{1}$$

$$\dot{\mathbf{v}} = \mathbf{M}^{-1}(\tau + \tau_{\text{env}}) - \mathbf{M}^{-1}(\mathbf{C}(\mathbf{v}) + \mathbf{D}(\mathbf{v}))\mathbf{v} \tag{2}$$

where $\mathbf{x} = [x \ y \ \psi]^T \in \mathbb{R}^{3\times 1}$ is the position and heading angle of the vessel in the inertial frame; $\mathbf{v} = [u \ v \ r]^T \in \mathbb{R}^{3\times 1}$ denotes the vessel velocity, which contains the surge velocity (u), sway velocity (v), and yaw rate (r) in the body fixed frame; $\mathbf{T}(\mathbf{x}) \in \mathbb{R}^{3\times 3}$ is the transformation matrix converting a state vector from body frame to inertial frame; $\mathbf{M} \in \mathbb{R}^{3\times 3}$ is the positive-definite symmetric added mass and inertia matrix; $\mathbf{C}(\mathbf{v}) \in \mathbb{R}^{3\times 3}$ is the skew-symmetric vehicle matrix of Coriolis and centripetal terms; $\tau_{\text{env}} \in \mathbb{R}^{3\times 1}$ is the environmental disturbances from the wind, currents and waves; $\mathbf{D}(\mathbf{v}) \in \mathbb{R}^{3\times 3}$ is the positive-semi-definite drag matrix-valued function; $\tau = [\tau_u \ \tau_v \ \tau_r]^T \in \mathbb{R}^{3\times 1}$ is the force and torque applied to the vessel in all three DOFs, which is defined as follow

$$\tau = \mathbf{B}\mathbf{u} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \frac{a}{2} & -\frac{a}{2} & \frac{b}{2} & -\frac{b}{2} \end{bmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} \tag{3}$$

where $\mathbf{B} \in \mathbb{R}^{4\times3}$ is the control matrix describing the thruster configuration and $\mathbf{u} = [f_1\ f_2\ f_3\ f_4]^T \in \mathbb{R}^{4\times1}$ is the control vector where f_1, f_2, f_3 and f_4 represent the left, right, anterior, and rear thrusters, respectively; a is the distance between the transverse propellers and b is the distance between the longitudinal propellers. $\mathbf{M}$, $\mathbf{C}(\mathbf{v})$ and $\mathbf{D}(\mathbf{v})$ are mathematically described as follows:

$$\mathbf{M} = \text{diag}\{m_{11}, m_{22}, m_{33}\} \tag{4}$$

$$\mathbf{C}(\mathbf{v}) = \begin{bmatrix} 0 & 0 & -m_{22}v \\ 0 & 0 & m_{11}u \\ m_{22}v & -m_{11}u & 0 \end{bmatrix} \tag{5}$$

$$\mathbf{D}(\mathbf{v}) = \text{diag}\{X_u, Y_v, N_r\} \tag{6}$$

Further, by combining (1) and (2), the complete dynamic model of the vessel is reformulated as follow

$$\dot{\mathbf{q}}(t) = f(\mathbf{q}(t), \mathbf{u}(t)) \tag{7}$$

where $\mathbf{q} = [x\ y\ \psi\ u\ v\ r]^T \in \mathbb{R}^{6\times1}$ is the state vector of the vessel, and $f(\cdot,\cdot,\cdot) : \mathbb{R}^{n_q} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \longrightarrow \mathbb{R}^{n_q}$ denote the continuously differentiable state update function. The system model describes how the full state $\mathbf{q}$ changes in response to applied control input $\mathbf{u} \in \mathbb{R}^{4\times1}$. Similarly, a nonlinear measurement model denoted $\mathbf{h}(t)$ can be described with the following equation:

$$\mathbf{z}(t) = h(\mathbf{q}(t), \mathbf{u}(t)) \tag{8}$$

where $\mathbf{z} = [x\ y\ \psi\ r\ f_1\ f_2\ f_3\ f_4] \in \mathbb{R}^{8\times1}$ denotes the measurement vector, and $h(\cdot,\cdot) : \mathbb{R}^{n_q} \times \mathbb{R}^{n_u} \longrightarrow \mathbb{R}^{n_z}$ denote measurement function.

Next, the unknown hydrodynamic parameter vector, $\xi = [m_{11}\ m_{22}\ m_{33}\ X_u\ Y_v\ N_r]^T$, in the dynamical model is required to be identified before applying it to the controller. The estimation of ξ using the experimental data set $\mathbf{v}^s, \mathbf{u}^s$ is a grey-box identification problem. The identification can be treated as an optimization problem described below

$$\min_{\xi} \sum_{t=0}^{T_s} \varepsilon(t)^T w \varepsilon(t), \tag{9a}$$

$$\text{s.t.} \quad \xi_l \le \xi \le \xi_u, \tag{9b}$$

$$\mathbf{q}(t) = f(\mathbf{q}(t), \mathbf{u}(t), \xi), t \in [0\ T_s], \tag{9c}$$

where $\varepsilon(t) \in \mathbb{R}^{3\times1}$ is the deviation between the experimental velocity $\mathbf{v}^s(t)$ and the simulated velocity $\mathbf{v}^m(t)$ at time t. ξ_l and ξ_u are the lower and upper bounds of ξ, respectively. $w \in \mathbb{R}^{3\times3}$ represents the weight matrix for the optimization. Different from our previous work [17], we adopt Sequential Quadratic Programming (SQP) method to numerically solve (9) in this study because SQP satisfies bounds at all iterations and (9) is not a large-scale optimization problem.

B. Nonlinear Model Predictive Control

For our trajectory tracking problem involving online dynamics learning, we formulate the optimal control problem for NMPC in the form of a least square function to penalize the deviations of predicted state ($\mathbf{q}_k$) and control ($\mathbf{u}_k$) trajectories from their specified references, over the given prediction horizon window N_c ($t_j \le t \le t_{j+N_c}$):

$$\min_{\mathbf{q}_k, \mathbf{u}_k} \frac{1}{2} \Big\{ \sum_{k=j}^{j+N_c-1} (\|\mathbf{q}_k - \mathbf{q}_k^{\text{ref}}\|^2_{W_q} + \|\mathbf{u}_k - \mathbf{u}_k^{\text{ref}}\|^2_{W_u}) + \|\mathbf{q}_{N_c} - \mathbf{q}_{N_c}^{\text{ref}}\|^2_{W_{N_c}} \Big\} \tag{10a}$$

$$\text{s.t.} \quad \mathbf{q}_j = \hat{\mathbf{q}}_j, \tag{10b}$$

$$\mathbf{q}_{k+1} = f(\mathbf{q}_k, \mathbf{u}_k), k = j, \cdots, j+N_c-1, \tag{10c}$$

$$\mathbf{q}_{k,\min} \le \mathbf{q}_k \le \mathbf{q}_{k,\max}, k = j, \cdots, j+N_c, \tag{10d}$$

$$\mathbf{u}_{k,\min} \le \mathbf{u}_k \le \mathbf{u}_{k,\max}, k = j, \cdots, j+N_c-1, \tag{10e}$$

where $\mathbf{q}_k \in \mathbb{R}^{n_q}$ denotes the vessel state, $\mathbf{u}_k \in \mathbb{R}^{n_u}$ denotes the control input, $\hat{\mathbf{q}}_j \in \mathbb{R}^{n_q}$ denotes the current state estimate, $\mathbf{q}_k^{\text{ref}}$ and $\mathbf{u}_k^{\text{ref}}$ denote the time-varying state and control references, respectively; $\mathbf{q}_{N_c}^{\text{ref}}$ denotes the terminal state reference; $W_q \in \mathbb{R}^{n_q\times n_q}$, $W_u \in \mathbb{R}^{n_u\times n_u}$, and $W_{N_c} \in \mathbb{R}^{n_q\times n_q}$ are the positive definite weight matrices that penalize deviations from the desired values. These weight matrices are assumed constant for a certain scale vessel in this study. Moreover, $\mathbf{p}$ is the parameter vector in the model which is referred as the payload of the vessel in this study. Furthermore, $\mathbf{q}_{k,\min}$ and $\mathbf{q}_{k,\max}$ denote the lower and upper bounds of the states, respectively; $\mathbf{u}_{k,\min}$ and $\mathbf{u}_{k,\max}$ denote the lower and upper bounds of the control input, respectively.

The weighting matrices W_q, W_u and W_{N_c} for the NMPC used in the experiments are selected as

$$W_q = \text{diag}\{200, 200, 100, 10, 10, 10\} \tag{11}$$

$$W_{N_c} = \text{diag}\{1000, 1000, 500, 50, 50, 150\} \tag{12}$$

$$W_u = \text{diag}\{1, 1, 1, 1\} \tag{13}$$

The prediction horizon $N_c = 4$ s, and the constraints on the control input $\mathbf{u}$ used in the experiments are chosen as follow

$$-50_{4\times1}\ \text{N} \le \mathbf{u}_{\text{quart}} \le 50_{4\times1}\ \text{N} \tag{14}$$

C. Nonlinear Moving Horizon Estimation

Online state estimation is employed to further generate more accurate states considering the dynamics and state constraints of the vessel. Nonlinear Moving Horizon Estimation (NMHE) is an online optimization-based state estimation approach that can handle nonlinear systems and satisfy inequality constraints on the estimated states and parameters [21]. In a similar manner, we utilize a least square function to penalize the deviation of estimated outputs from the measurements, formulating the NMHE problem as follows: At current time t_j there shall be N_c measurements $\mathbf{z}_{j-M+1}, ..., \mathbf{z}_j \in \mathbb{R}^{n_z}$ available, associated to the time instants $t_{j-N_c+1} < ... < t_j$ in the past. $T_{\text{E}} = t_j - t_{j-N_c+1}$ is the length of the horizon. Finally, the discrete time dynamic optimization problem to estimate the constrained states ($\hat{\mathbf{q}}$) at time t_j using the available measurements within the horizon, is solved by

$$\min_{\hat{\mathbf{q}}_k} \|\hat{\mathbf{q}}_{j-N_c+1} - \bar{\mathbf{q}}_{j-N_c+1})\|^2_{P_L} + \sum_{k=j-N_c+1}^{j} \|\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{q}}_k, \mathbf{u}_k)\|^2_{R_T} \quad (15a)$$

$$\text{s.t. } \hat{\mathbf{q}}_{k+1} = f(\hat{\mathbf{q}}_{k+1}, \mathbf{u}_k) + \mathbf{w}_k, k = j-N_c+1, ..., j-1, \quad (15b)$$

$$\hat{\mathbf{q}}_{k,\min} \leq \hat{\mathbf{q}}_k \leq \hat{\mathbf{q}}_{k,\max}, k = j-N_c+1, ..., j, \quad (15c)$$

where $\hat{\mathbf{q}}_{k,\min}$ and $\hat{\mathbf{q}}_{k,\max}$ denote the lower and upper bounds on the estimated states of the vessel, respectively. The weighting matrices P_{N_c} and R_k are usually interpreted as the inverses of the measurement and process noise covariance matrices, respectively. P_{N_c} and R_k should be selected adequately to achieve good state estimates based on the knowledge or prediction of the error distributions. Moreover, $\bar{\mathbf{q}}_{j-N_c+1}$ represents the estimated state at the start of estimation horizon t_{j-N_c+1}.

The weighting matrices P_L and R_T are the inverses of the process and measurement noise covariance matrices, respectively. Considering the noise characteristics of sensors listed in Section II-B, the weighting matrix R_T used in the experiments is chosen as follow

$$\begin{aligned} R_T &= \text{diag}\{\sigma_x^2, \sigma_y^2, \sigma_\psi^2, \sigma_r^2, \sigma_{f_1}^2, \sigma_{f_2}^2, \sigma_{f_3}^2, \sigma_{f_4}^2\}^{-1} \\ &= \text{diag}\{0.0005, 0.0005, 0.0005, 0.0001, 1, 1, 1, 1\}^{-1} \end{aligned} \quad (16)$$

Moreover, the weighting matrix P_L used in the experiments are selected as follow

$$\begin{aligned} P_L &= \text{diag}\{\sigma_x^2, \sigma_y^2, \sigma_\psi^2, \sigma_u^2, \sigma_v^2, \sigma_r^2\}^{-1} \\ &= \text{diag}\{1, 1, 1, 0.1, 0.1, 1\}^{-1} \end{aligned} \quad (17)$$

V. Simultaneous Localization and Mapping

An overview of the proposed SLAM system, which is adapted from [22], is shown in Fig. 6. The system receives sensor data from a 3D lidar, an IMU, and a GPS. SLAM algorithm aims to estimate the state $\mathbf{x} = [x\ y\ \psi] \in \mathbb{R}^{3\times 1}$ of the vessel given the sensor measurements. This state estimation problem can be formulated as a posteriori (MAP) problem. To seamlessly incorporate measurements from various sensors, we utilize a factor graph to model this problem. Then solving the MAP problem is equivalent to solving a nonlinear least-squares problem [23].

We introduce three types of *factors* along with one *variable* type for factor graph construction. This variable, representing the vessel's state, is referred to as *nodes* of the graph. The three types of factors are: (a) lidar odometry factors, (b) GPS factors, and (c) loop closure factors. To limit the memory usage and improve the efficiency of the localization system, we add new node $\mathbf{x} \in \mathbb{R}^{3\times 1}$ to the graph using a simple but effective heuristic approach. A new node $\mathbf{x}$ is only added when the position or rotation change of the vessel exceeds a user-defined threshold. We use incremental smoothing and mapping with the Bayes tree (iSAM2) [24] to optimize the factor graph upon the insertion of a new node. The process for generating the aforementioned factors is described in the following sections.

A. Lidar Odometry Factor

We perform feature extraction using the raw point cloud from the 3D lidar. Similar to the process introduced in [25], we first project the raw point cloud onto a range image as the arriving point cloud may not be organized. Each point in the point cloud is associated with a pixel in the range image. Then we calculate the roughness value of a pixel in the range image using its neighboring range values. Points with a large roughness value are classified as edge features. Similarly, a planar feature is determined by a small roughness value. We denote the extracted edge and planar features from a lidar scan at time t as F_t^e and F_t^p respectively. All the features extracted at time t compose a lidar *frame* $\mathbb{F}_t$, where $\mathbb{F}_t = \{\mathrm{F}_t^e, \mathrm{F}_t^p\}$. Note that $\mathbb{F}_t$ is represented in the local sensor frame. A more detailed description of the feature extraction process can be found in [25].

It is computationally intractable to add factors for every lidar frame. Thus we adopt the concept of *keyframe* selection, which is widely used in the visual SLAM field. We select a lidar frame $\mathbb{F}_{t+1}$ as a keyframe when the vessel's pose change exceeds a user-defined threshold. In this paper, we select a new lidar keyframe when the vessel's position change exceeds $1m$ or the rotation change exceeds 10°. The lidar frames between two keyframes are discarded. When a lidar keyframe is selected, we perform scan-matching to calculate the relative transformation between the new keyframe with the previous sub-keyframes. The sub-keyframes are obtained by transforming the previous n keyframes into the global world frame. These transformed keyframes are then merged together into a voxel map $\mathbf{M}_t$. Note that $\mathbf{M}_t$ is composed of two sub-voxel maps because we extract two types of features in the previous step. These two sub-voxel maps are denoted as $\mathbf{M}_t^e$, the edge feature voxel map, and $\mathbf{M}_t^p$, the planar feature voxel map. The relationships between the feature sets and the voxel maps can be represented as:

$$\begin{aligned} &\mathbf{M}_t = \{\mathbf{M}_t^e, \mathbf{M}_t^p\} \\ &where: \mathbf{M}_t^e = {}'\mathrm{F}_t^e \cup' \mathrm{F}_{t-1}^e \cup ... \cup' \mathrm{F}_{t-n}^e \\ &\qquad\quad \mathbf{M}_t^p = {}'\mathrm{F}_t^p \cup' \mathrm{F}_{t-1}^p \cup ... \cup' \mathrm{F}_{t-n}^p, \end{aligned}$$

where ${}'\mathrm{F}_t^e$ and ${}'\mathrm{F}_t^p$ are the transformed edge and planar features in the global frame. n is chosen to be 25 for all the tests. The new lidar keyframe $\mathbb{F}_{t+1}$ is transformed into the global world frame using the initial guess from IMU. The transformed new keyframe in the world frame is denoted as ${}'\mathbb{F}_{t+1}$. With ${}'\mathbb{F}_{t+1}$ and the voxel map $\mathbf{M}_t$, we perform scan-matching using the method proposed in [26] and obtain the relative transformation $\Delta\mathbf{T}_{t,t+1}$ between them. At last, $\Delta\mathbf{T}_{t,t+1}$ is added as the lidar odometry factor into the factor graph.

B. GPS Factor

Though we can achieve low-drift state estimation solely by using lidar odometry factors, the localization system still suffers from drift during long-duration navigation tasks. We thus utilize the absolute measurements from GPS and incorporate them as GPS factors into the factor graph. When

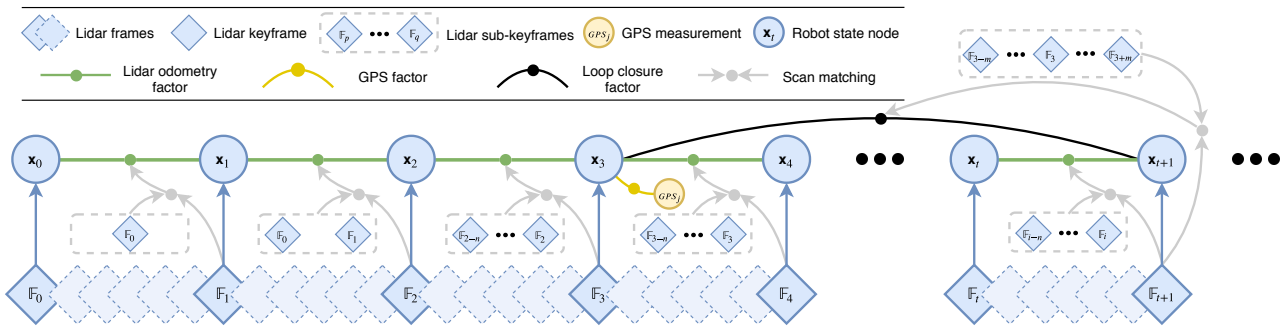

Fig. 6. The framework of the SLAM system. The system receives input from a 3D lidar, an IMU, and a GPS. Three types of factors are introduced to construct the factor graph.: (a) lidar odometry factor, (c) GPS factor, and (c) loop closure factor. The generation of these factors is discussed in Section V.

the GPS measurements are available, we first transform them into the Cartesian coordinate frame using the method proposed in [27]. Upon the insertion of a new node to the factor graph, we then add a new GPS factor and associate it with the new node.

C. Loop Closure Factor

Due to the utilization of a factor graph, loop closures can also be seamlessly incorporated into the proposed system. Successful detection of loop closure is introduced as a loop closure factor in the factor graph. For illustration, we introduce a naive but effective loop closure detection approach that is based on Euclidean distance. When a new state $\mathbf{x}_{i+1}$ is added into the factor graph, as is shown in Fig. 6, we search the graph and find the prior states that are within a certain distance to $\mathbf{x}_{i+1}$. For example, $\mathbf{x}_3$ is the returned candidate state. We then extract sub-keyframes $\{\mathbb{F}_{3-m},...,\mathbb{F}_3,...,\mathbb{F}_{3+m}\}$ and merge them into a local voxel map, which is similar to the voxel map introduced in Sec. V-A. If a successful match can be found between $\mathbb{F}_{i+1}$ and the voxel map, we obtain the relative transformation $\Delta\mathbf{T}_{3,\,i+1}$ and add it as a loop closure factor to the graph. Throughout the map, we choose m to be 12, and the search distance for loop closures is set to be 15 m. In practice, we find adding loop closure factors is especially useful for eliminating the drift when conducting navigation tasks in GPS-denied regions.

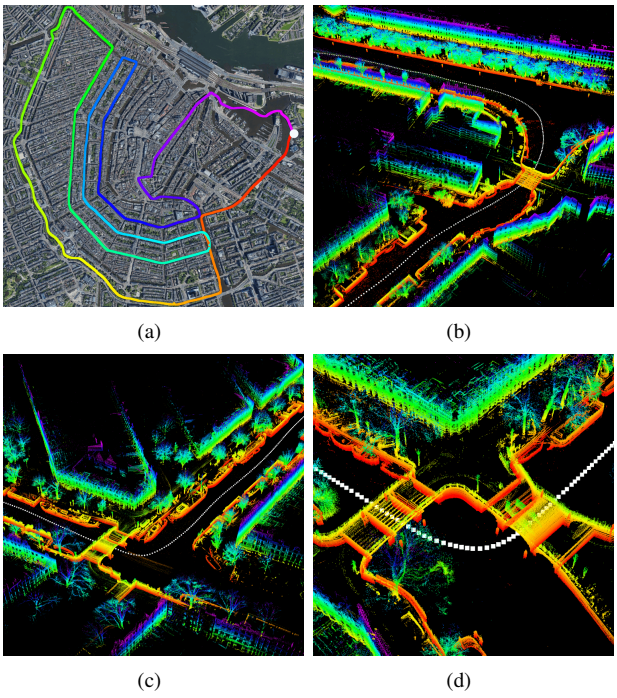

Fig. 7. The estimated trajectory (a) and representative point cloud maps (b-d) of the proposed SLAM system using a dateset gathered in Amsterdam, the Netherlands. In (a), the white dot marks the start and end location. Trajectory color variation indicates the elapse of time. The trajectory direction is clockwise. In (b-d), color variation indicates elevation change.

VI. Experiments and Results

We performed several experiments in canals and rivers to verify the algorithms as well as demonstrate the effectiveness of the developed autonomous system. All the algorithms including SLAM, NMPC, NMHE, path planner are executed on an onboard computer (described in Section II-B). These algorithms are updated at a rate of 10 Hz.

A. SLAM Results

To test the performance of the proposed SLAM system, we mount the sensor suite, which includes a Lidar, an IMU, and a GPS, on a manned boat and cruised along the canals of Amsterdam for 3 hours. We start and finish the data-gathering process at the same location. Altogether, we collected 107,656 lidar scans with an estimated trajectory length of 19,065 m. Performing SLAM in the canal environment is challenging due to several reasons. Many bridges over the canals pose degenerate scenarios, as there are few useful features when the boat is under them, similar to moving through a long, featureless corridor. Bridges and buildings obstruct the reception of GPS data and result in intermittent GPS availability throughout the dataset. We also observe numerous false directions from the lidar when direct sunlight is in the sensor field-of-view.

We test the performance of the system by disabling the insertion of GPS factor, loop closure factor, or both. When we solely use lidar odometry factors with two other factors being disabled, the system fails to produce meaningful results and suffers from great drift in a featureless environment. When we use both lidar odometry factors and GPS factors, the system can provide accurate pose estimation in the horizontal plane while suffers from drift in the vertical direction.

This is because the elevation measurement from GPS is very inaccurate - giving rise to altitude errors approximating 100 m in our tests, which further motivates our usage of loop closure factors. Upon finishing processing the dataset, the proposed SLAM system achieves a relative translation error of 0.17 m when returning to the start location. We also note that the average runtime for registering a new lidar keyframe and optimizing the graph is only 79.3 ms, which is suitable for deploying the proposed system on low-powered embedded hardware.

B. Results of NMPC and NMHE

First, we experimentally estimated the dynamics model of the vessel to achieve superior performance of NMPC for tracking control. The data was gathered when the vessel was remotely controlled to perform sinusoidal movements in Charles River (Fig. 8), which couples the surge, sway, and rotation motions. The input force and the vessel velocity were

(a) (b)

Fig. 8. Testing Roboat II in Charles River. (a) Close-up shot; (b) long shot.

recorded at a rate of 10 Hz in the experiments. The trials were repeated five times. The duration of each trial was around 150 s. We utilized the optimization algorithm described in Section IV-A and identified the hydrodynamic parameters ξ for Roboat II as follows: $m_{11} = 172$ kg, $m_{22} = 188$ kg, $m_{33} = 24$ kg$\cdot$m^2, $X_u = 38$ kg/s, $Y_v = 168$ kg/s and $N_r = 16$ kg$\cdot$m^2/s.

Next, we tested the performance of the NMPC tracking and the NMHE state estimation on Roboat II in Charles River. We use an $A^\star$ planner to generate the reference paths for NMPC. The experiment was conducted as follows. First, we assigned a goal point for the planner using a ROS graphical interface, RVIZ. The planner will generate a feasible path starting from the current position and ending with the goal position of the vessel. Then, Roboat II starts to track the reference trajectories to approach the destination. After Roboat II reached the destination, we selected another goal for Roboat II which initiates a new tracking section. We repeated the same assignments several times in the experiments. NMHE was running at the same time to estimate the states of Roboat II during the experiments.

Trajectory and heading angle tracking performances of Roboat II are shown in Fig. 9. This demonstrates that the MPC can successfully track on the references even with

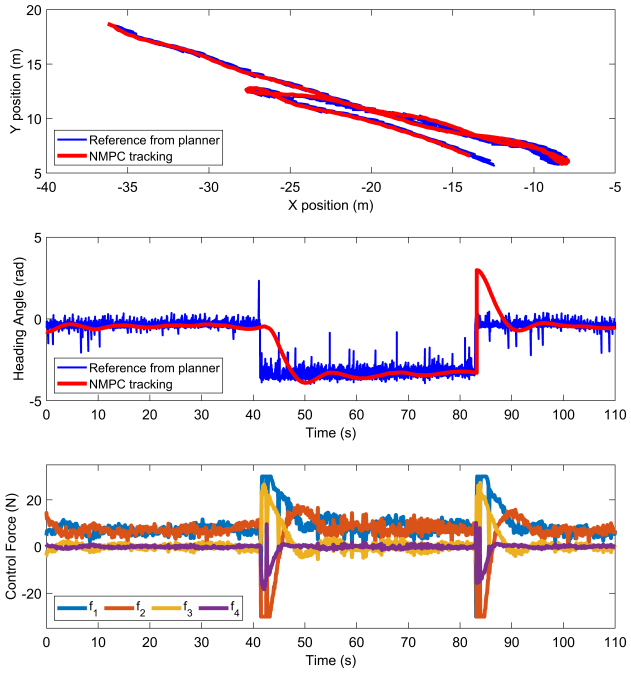

Fig. 9. Performance of the NMPC tracking in the river test. (a) trajectory; (b)heading angle; (c) forces.

environmental noises in the river. The position tracking RMSE (Root Mean Square Error) value for NMPC on Roboat II is 0.096 m while the heading angle tracking RMSE value for NMPC is 0.149 rad. Note that we penalize (the weight coefficient) less for the heading angle than that of the positions in (10a). The reason is that we focus more on tracking the desired path accurately. Moreover, less penalization on the heading angle avoids oscillations around the desired path. The simple planner generates the whole reference path to the goal position at the rate of 10 Hz. In Fig. 9(a) and (b), we only show 10 reference points starting from the current position of Roboat II because the generated reference path is always slightly oscillating. A more careful inspection from Fig. 9(b) indicates that the noisy reference heading angle at each time instant during the experiments.

The control forces for Roboat II are shown in Fig.9(a). These generated control signals by the NMPC is restrained within the lower and upper bounds specified in (14) in Section IV-B. It is clear that the left and right thrusters contribute significantly as the system in on-track. If the system was not on-track, all the four thrusters would contribute significantly (at around 42 s and 84 s) to help the system rapidly reach the reference positions and orientations.

The indirectly measured and estimated linear and angular velocities for Roboat II are shown in Fig. 10. This demonstrates that the MHE can successfully deal with noises on the measurements. The reference for linear longitudinal velocity u and lateral velocity v are respectively set to 0.6 m/s and 0 m/s throughout the reference path generation. The estimates always track around these reference values as the system is on-track, which also indicates the effectiveness of the NMPC tracking. The peak and valley areas in the velocities

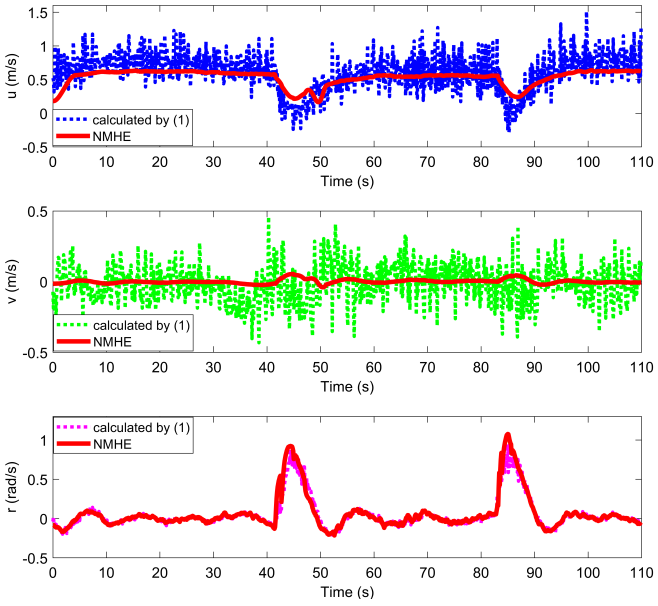

Fig. 10. Performance of the NMHE state estimation in the river test. (a) linear speed u; (b)linear speed v; (c)angular speed r.

suggest the process that Roboat II approaches the current goal position and heads to the next goal by fast turning itself.

VII. Conclusion and Future Work

In this paper, we have developed a novel autonomous surface vessel (ASV), called Roboat II, for urban transportation. Roboat II is capable of performing accurate simultaneous localization and mapping (SLAM), receding horizon tracking control and estimation, and path planning in the urban waterways.

Our work will be extended in the following directions in the near future. First, we will explore more efficient planning algorithms to enable the vessel to handle complicated scenarios in the waterways. Second, we will apply active object detection and identification to improve Roboat's understanding of its environment for robust navigation. Third, we will estimate disturbances such as currents and waves to further improve the tracking performance in more noisy waters. Fourth, we will develop algorithms for multi-robot formation control and self-assembly on the water, enabling the construction of on-demand large-scale infrastructure.

References

[1] P. Corke, C. Detweiler, M. Dunbabin, M. Hamilton, D. Rus, and I. Vasilescu, "Experiments with underwater robot localization and tracking," in *Proc. 2007 IEEE Int. Conf. Robot. Autom.*, 2007, pp. 4556–4561.

[2] M. Doniec, I. Vasilescu, C. Detweiler, and D. Rus, "Complete SE3 underwater robot control with arbitrary thruster configurations," in *Proc. 2010 IEEE Int. Conf. Robot. Autom.*, 2010, pp. 5295–5301.

[3] J. Curcio, J. Leonard, and A. Patrikalakis, "SCOUT - a low cost autonomous surface platform for research in cooperative autonomy," in *Proc. MTS/IEEE Oceans*, 2005, pp. 725–729.

[4] L. Paull, S. Saeedi, M. Seto, and H. Li, "AUV navigation and localization: A review," *IEEE J. Ocean. Eng.*, vol. 39, no. 1, pp. 131–149, 2014.

[5] A. Dhariwal and G. S. Sukhatme, "Experiments in robotic boat localization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 2007, pp. 1702–1708.

[6] Z. Liu, Y. Zhang, X. Yu, and C. Yuan, "Unmanned surface vehicles: An overview of developments and challenges," *Annu Rev Control*, vol. 41, pp. 71 – 93, 2016.

[7] H. K. Heidarsson and G. S. Sukhatme, "Obstacle detection and avoidance for an autonomous surface vehicle using a profiling sonar," in *Proc. 2011 IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 731–736.

[8] K.-C. Ma, L. Liu, H. K. Heidarsson, and G. S. Sukhatme, "Data-driven learning and planning for environmental sampling," *Journal of Field Robotics*, vol. 35, no. 5, pp. 643–661.

[9] T. Shan, W. Wang, B. Englot, C. Ratti, and D. Rus, "A Receding Horizon Multi-Objective Planner for Autonomous Surface Vehicles in Urban Waterways," *arXiv preprint arXiv:2007.08362*, 2020.

[10] W. B. Klinger, I. R. Bertaska, K. D. von Ellenrieder, and M. R. Dhanak, "Control of an unmanned surface vehicle with uncertain displacement and drag," *IEEE J Ocean Eng*, vol. 42, no. 2, pp. 458–476, 2017.

[11] B. J. Guerreiro, C. Silvestre, R. Cunha, and A. Pascoal, "Trajectory tracking nonlinear model predictive control for autonomous surface craft," *IEEE Trans Control Syst Technol*, vol. 22, no. 6, pp. 2160–2175, 2014.

[12] J. E. Manley, "Unmanned surface vehicles, 15 years of development," in *Proc. MTS/IEEE Oceans*, Sept 2008, pp. 1–4.

[13] W. Naeem, R. Sutton, and T. Xu, "An integrated multi-sensor data fusion algorithm and autopilot implementation in an uninhabited surface craft," *OCEAN ENG*, vol. 39, pp. 43–52, 2012.

[14] H. Ashrafiuon, K. R. Muske, L. C. McNinch, and R. A. Soltan, "Sliding-mode tracking control of surface vessels," *IEEE Trans. Ind. Electron*, vol. 55, no. 11, pp. 4004–4012, 2008.

[15] H. K. Khalil, "Noninear systems," *Prentice-Hall, New Jersey*, vol. 2, no. 5, pp. 5–1, 1996.

[16] R. Skjetne, N. Smogeli, and T. I. Fossen, "A nonlinear ship manoeuvering model: Identification and adaptive control with experiments for a model ship," *Model. Ident. Control*, vol. 25, no. 1, p. 3, 2004.

[17] W. Wang, L. Mateos, S. Park, P. Leoni, B. Gheneti, F. Duarte, C. Ratti, and D. Rus, "Design, modeling, and nonlinear model predictive tracking control of a novel autonomous surface vehicle," in *Proc. 2018 IEEE Int. Conf. Robot. Autom*, 2018, pp. 6189–6196.

[18] W. Wang, B. Gheneti, L. A. Mateos, F. Duarte, C. Ratti, and D. Rus, "Roboat: An autonomous surface vehicle for urban waterways," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2019, pp. 6340–6347.

[19] L. A. Mateos, W. Wang, B. Gheneti, F. Duarte, C. Ratti, and D. Rus, "Autonomous latching system for robotic boats," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 7933–7939.

[20] W. Wang, L. Mateos, Z. Wang, K. W. Huang, M. Schwager, C. Ratti, and D. Rus, "Cooperative control of an autonomous floating modular structure without communication: Extended abstract," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2019, pp. 44–46.

[21] C. V. Rao, J. B. Rawlings, and D. Q. Mayne, "Constrained state estimation for nonlinear discrete-time systems: stability and moving horizon approximations," *IEEE Transactions on Automatic Control*, vol. 48, no. 2, pp. 246–258, Feb 2003.

[22] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping," *arXiv preprint arXiv:2007.00258*, 2020.

[23] F. Dellaert, M. Kaess *et al.*, "Factor Graphs for Robot Perception," *Foundations and Trends® in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017.

[24] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental Smoothing and Mapping using the Bayes Tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[25] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4758–4765.

[26] J. Zhang and S. Singh, "Low-drift and Real-time Lidar Odometry and Mapping," *Autonomous Robots*, vol. 41, no. 2, pp. 401–416, 2017.

[27] T. Moore and D. Stouch, "A Generalized Extended Kalman Filter Implementation for the Robot Operating System," in *Intelligent autonomous systems 13*. Springer, 2016, pp. 335–348.