2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Log Loading Automation for Timber-Harvesting Industry

Elie Ayoub[1], Heshan Fernando[2], William Larrivée-Hardy[2], Nicolas Lemieux[2], Philippe Giguère[3], Inna Sharf[1,2]

***Abstract*— The timber-harvesting industry is lagging its peer industries, such as mining and agriculture, with respect to deployment of robotic, AI and autonomous technologies. In this paper, we tackle automation of a critical task that arises in transporting logs from the forest to the sawmill: the log loading operation. This work is motivated by the acute shortages of human operators and the need to improve the efficiencies of timber-harvesting processes. To this end, we demonstrate the full autonomy pipeline for the log loading operation with a fixed-base manipulator (a.k.a., the crane), starting with perception of logs around the machine, then grasp planning for where to grasp logs, through motion planning and control of the log loading maneuver. Our main contribution is in the full integration of the necessary elements to achieve a completely autonomous loading cycle, where the crane picks up and loads all logs within its reach on a trailer. Notable features of our implementation are a generalizable perception stack, a grasp planner to pick up multiple logs at a time and an extensive experimental campaign conducted outdoors, on a commercial log loader retrofitted for autonomy. Our results demonstrate an overall 87% success rate of the log loading operation, with primary failure cases due to log segmentation errors and deficiencies in the final height adjustment algorithm for grasping logs. We also present detailed timing results of the main parts of the autonomy pipeline, which support the feasibility of deployment in operational environment.**

I. Introduction

A. Background and Motivation

Interest in applying advances in robotics and AI in the forestry industry and, specifically for forestry operations, has increased significantly over the past few years [1], [2]. Research on automating the operation of timber harvesting machines, in particular, is primarily motivated by the lack of skilled machine operators and the need to increase efficiencies. These factors make the shift from human-operated to automated forestry machines an urgent concern [3], [4]. While forestry operations are numerous and diverse, the log loading operation is critical to the wood supply chain as it is carried out at several points along the supply route, from the forest to the sawmill. Due to the variability in operational scenarios and unstructured nature of the operating environment, it is a challenging task to automate and forms the central focus of this paper.

To date, efforts have been made to assist operators of forestry machines and to enhance human-machine cooperation through automating some decisions for the operator [5] or enabling teleoperation [6]. Few works aim to achieve complete autonomy for tree felling and log loading operations. For the log loading task, several works in existing literature address the problem of path planning for the log-loader's crane [7], [8]. Some researchers have addressed autonomous log grasping [8]–[11]; however, there are no results to date demonstrating a complete and fully autonomous log loading operation, comprising multiple loading maneuvers from arbitrary pile arrangements around the machine.

Fig. 1: FPInnovations log-loading testbed with crane in home configuration

B. About this Paper

Motivated by the above considerations, our industry partner, FPInnovations, had initiated research on log loading automation in 2020. The goal of this paper is to present results achieved to date, highlighting the novel aspects of the work and our outlook for the future. In particular, the principal contributions of this work are:

- the development and implementation of a complete autonomous pipeline to execute a full log loading cycle with a fixed-base log loader;
- experimental results demonstrating performance of the autonomous pipeline, with an open discussion of its current limitations. The results include timing statistics for execution of different parts of the pipeline in order to draw conclusions on the competitiveness of autonomous log loading operations in comparison to execution by human operators.

[1]The co-author is with Department of Mechanical Engineering, McGill University, Montreal, Canada elie.ayoub@mail.mcgill.ca
[2]The co-author is with FPInnovations, Pointe-Claire, QC H9R 3J9, Canada firstname.lastname@fpinnovations.ca
[3]The co-author is with Computer Science and Software Engineering Department, Université Laval, Quebec, Canada Philippe.Giguere@ift.ulaval.ca

The rest of this paper is divided as follow. In section II, we describe the relevant theoretical details and the implementation of the fundamental building blocks of our autonomy pipeline developed for the crane testbed at FPInnovations, shown in Fig. 1. The testbed itself, including the control architecture employed to implement the autonomous pipeline, is presented in section III. Performance results of the experimental trials on autonomous log loading carried out with the testbed are summarized and discussed in section IV, followed by conclusions in section V.

II. Autonomy Building Blocks For Log Loading

A. Overview

Although most machines employed for loading logs are mobile, the operation itself is carried out with a crane—an articulated arm of the machine—while the base remains stationary. From the robotics perspective, the log loading operation is thus a pick-and-place task. It involves point-to-point maneuvering of the crane, with obstacle avoidance. The more challenging aspects of automating log loading are: (i) detection of logs to be loaded, which falls under the umbrella of perception and (ii) grasp planning which, aside from the trivial case of a single isolated log, requires determining where to grasp multiple logs in a pile. For this reason, two of the main blocks in our log loading autonomy pipeline depicted in Fig. 2 will address these issues. Another key component of our architecture is the motion planner, responsible for generating joint trajectories to reposition the crane. These three components will be described below.

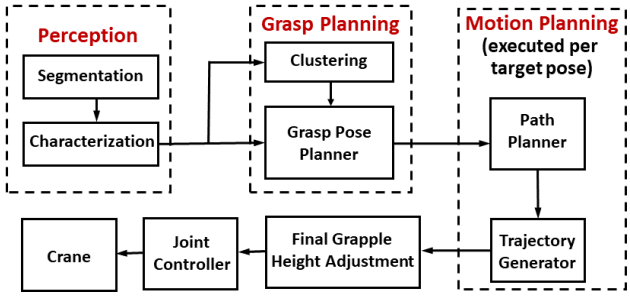


Fig. 2: Autonomy pipeline for logs loading maneuver. The perception and grasp planning blocks are executed at the start of a maneuver. Final grapple adjustment is only executed before the approach to the target log(s).

B. Perception: Log Detection and Characterization

The perception problem for the log loading task requires, first and foremost, detection of logs in the vicinity of the machine. One reason why this is non-trivial is that logs, in forest or in mill yard, rarely present themselves in isolation. Instead, they are usually cluttered with other logs in various arrangements, ranging from mini-piles of several logs to large piles. This in turn results in partial occlusions with objects of similar appearance, which makes instance segmentation more challenging. Since forestry operations are performed throughout the year, the log detection solution must also be robust to weather and environmental conditions.

Our perception pipeline uses the log detector from [12], based on the Mask2Former architecture [13]. Whereas [14] uses bounding boxes to localize logs, we leverage a mask output to detect individual logs within the reach of the crane. Our implementation uses a Swin-B transformer backbone pre-trained on the ImageNet-22k dataset [15] and fine-tuned on the *TimberSeg 1.0* dataset [12]. It achieved 84.3% average precision with 0.5 intersection over union (IoU) and a recall of 65.2%. The TimberSeg dataset contains 220 images comprising 2500 segmented logs, along with their mask annotations. Importantly, only 37 images of this dataset were collected on the FPInnovations log loading testbed. The rest of the images were collected with dash-cams on log loaders operating in forests over several months, in a range of weather conditions. With the variety of environments in the dataset, we expect our perception pipeline to be robust to changes in the operating environment.

In some of the previous work on log loading automation [8], it is assumed that the log characteristics, namely its position, orientation and diameter, are known. In others [14] and similarly to us, a perception pipeline is responsible for this characterization. In our approach, the segmented log masks in one of the RGB stereo images are associated with their corresponding 3D point cloud to obtain positional information. To speed up processing, the segmented point cloud of each log is downsampled and analyzed with PCA to compute its principal directions. The ranges of points corresponding to the two principal directions are processed to estimate the log length and diameter, from which the center of the log is inferred. The principal directions are used to orient the log relative to the camera frame and then to transform the pose to the base frame of the machine. Fig. 3 shows a sample case of segmentation and log characterization, as seen from the camera on the log loading crane. We also note that since log characterization is a naturally parallelizable computation, it is executed in a different process for each log, taking advantage of the multiples cores of the CPU.

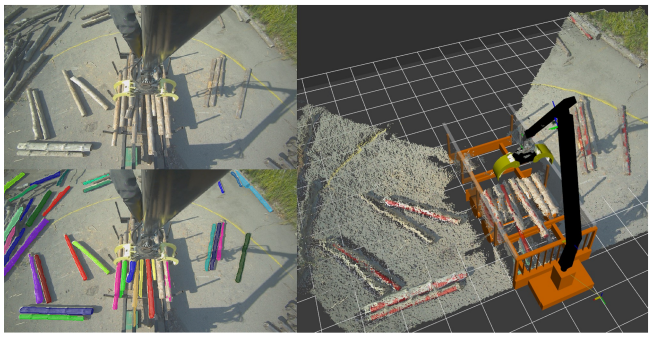


Fig. 3: Logs segmentation and characterization sample, FPInnovations log loading testbed: upper left panel—camera view, lower left panel—masked logs, right panel—overlay of characterized logs in red over point cloud data.

C. Grasp Planning

Although the literature on grasp planning is vast, very few works have addressed grasp planning in the context

of forestry log loading. The approach in [16] showcases the use of convolutional neural network (CNN) from [17] to perform object segmentation and to generate single-log grasp predictions. Another deep learning approach is that in [9] where RGB data is fed to a CNN that predicts good grasping candidates, again on individual logs only, in 2D space. A recent manuscript [11] proposes RL and visual-servoing based on a virtual camera to achieve grasping of multiple logs.

In our approach, we employ the grasp planning pipeline from [18] with the addition of a clustering component: once the logs in the scene have been identified and characterized by the perception module, they are categorized in separate piles from each other based on the distance between them. The Agglomerative Clustering algorithm from the scikit-learn[1] python library is employed. Using a maximum linkage criterion, the distance between two clusters is defined as the euclidean distance between the two furthest points of the clusters. For our case, a maximum distance threshold of 2 m is used. After all clusters have been formed, the one that is nearest to the crane base is chosen to be the target for grasp planning. Logs belonging to non-chosen clusters are ignored for the remainder of the grasp planning pipeline.

Given the chosen cluster, virtual logs in the form of cylinders with corresponding characteristics (lengths, diameters and poses) to the cluster's logs are instantiated in a Gazebo environment, such that the poses of the virtual logs expressed in the virtual world frame match those of the real logs with respect to the crane base frame. A virtual depth camera is then placed above the reproduced logs to capture the depth data that will serve as the input to our grasp planning CNN. Employing a virtual environment and camera for our input grasp planning data allows for capturing any desired viewpoint of the target log cluster, without having to move the real crane. In our case, this virtual camera is positioned 3 m above the geometric center of a bounding box around the virtual logs. In cases where logs are present at the edges of the virtual camera's depth image, the camera's height above the logs is raised in small increments until no logs remain too close to the depth image's border, thus ensuring that all logs are completely captured by the camera and are at the center of attention of the image.

The depth images collected in the virtual environment are then fed to a fully convolutional neural network that outputs an image-space grasp map:

$$\tilde{G} = (Q, \tilde{\Theta}) \in \mathbb{R}^{2\times\tilde{X}\times\tilde{Y}} \tag{1}$$

where Q is the pixelwise quality distribution assigning a scalar between 0 and 1 that indicates the chances of grasping success to every pixel of the input image of size $\tilde{X}\times\tilde{Y}$, and $\tilde{\Theta}$ is the pixelwise image-space grasping angle distribution indicating the required orientation of the grapple to perform a grasp at every pixel of the input image. The optimal image-space grasping location $(\tilde{x}^*, \tilde{y}^*)$ is then inferred from the quality distribution Q by locating the pixel coordinates with the highest quality value. As for the grasping angle, it is extracted from the grasping angle distribution $\tilde{\Theta}$ using the same pixel coordinates. Fig. 4 shows a virtual environment containing a sample cluster of logs, along with the CNN's grasping location prediction in image space. The determined image-space grasping pose is finally converted to a grasping pose in the crane base frame, via the camera frame. The resulting target grapple pose is passed to the motion planner to execute the grasp.

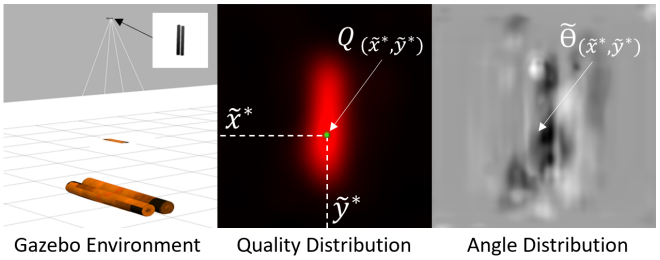


Fig. 4: Grasp planning algorithm stages: two log example

D. Motion Planning and Control

The motion planning block is responsible for generating first a path and then, the joint trajectories, to move the crane from its current position to the next specified target position. We employ the MoveIt[2] package (MoveIt 1 Noetic distribution), with the Open Motion Planning Library (OMPL) and the RRT* sampling-based algorithm to generate a collision free path in joint space. Self collision and three environmental obstacles are accounted for by MoveIt: the left and right (side) bounding boxes ('walls') of the trailer to circumscribe the side posts and, a 20 m $\times$ 20 m $\times$ 0.1 m box to define the ground plane. Our current implementation allocates a fixed time of 2 s to MoveIt to compute the trajectory plan. TRAC-IK (Inverse Kinematics) solver is used prior to MoveIt for transforming from task to joint space, when a new target pose is provided in cartesian space.

An individual loading maneuver is defined with a sequence of target poses, with MoveIt invoked at each one to plan to the subsequent target. The crane begins in its home configuration, with elbow joint high to ensure a good camera view of the area around the trailer (see Fig. 1). The maneuver includes five intermediate target positions, comprising the end-points of five rest-to-rest trajectory segments, as follows: first, from home position to 0.5 m above the grasp target pose, as per grasp planner; second, to the grasping height for closing the grapple, as per final grapple height adjustment; third, after grasping logs, to 0.5 m above the previous target; fourth, to 1.5 m above the fixed unload target position above the trailer and finally, to the unload target position where the grapple is commanded to fully open. The individual loading maneuver is completed by returning to the home position (sixth target), from which the maneuver is repeated until all logs around the trailer have been picked up.

It is worth noting that the maneuver sequence described above is not optimized in any sense, but was designed with

[1]https://scikit-learn.org/

[2]https://moveit.ros.org/

the primary goal of completing the task successfully. The fourth intermediate waypoint for the return path to the trailer was selected high enough to avoid collisions since, currently, the logs carried by the grapple are not modelled for obstacle avoidance.

In general, we found the logs grasping task to be tolerant to the *x*-*y* grapple positioning errors, but sensitive to the vertical (z) location of the grapple: positioning the grapple too low results in the tongs digging into the ground. Conversely, positioning the grapple too high may result in either a complete miss of the logs or a pinch grasp, which is undesirable. To address this problem, the final grapple height adjustment algorithm is employed (see Fig. 2), which uses point cloud data to provide a better estimate of the ground location. More specifically, once at the grapple target pose, three bounding boxes of pre-defined dimensions are created under the grapple's center, left tong tip and right tong tip, respectively. For each box, a vertical distance value towards the ground is computed based on the point cloud intersecting it, and the largest obtained distance value out of the three is used for the final grapple height adjustment.

Tracking of the planned trajectory is achieved with proportional position control and velocity feed-forward for the four boom positioning joints and the grapple orientation. Given the joint reference positions $\mathbf{q_r}$, reference velocities $\dot{\mathbf{q}}_\mathbf{r}$ and measured positions $\mathbf{q}$, the control outputs are calculated as:

$$\mathbf{u} = \mathbf{K_p}(\mathbf{q_r} - \mathbf{q}) + \mathbf{K_v}\dot{\mathbf{q}}_\mathbf{r}. \tag{2}$$

The controller does not take into account the crane's load capacity, nor any nonlinearities of the system. The control gains $\mathbf{K_p}$ and $\mathbf{K_v}$ were pre-tuned on the real system to achieve adequate tracking performance. The grapple opening was commanded with open-loop, bang-bang inputs.

III. Crane Testbed

A. Crane and Environment

To conduct the log-loading experiments presented in this paper, we employ the log loading testbed located in the outdoor parking lot of FPInnovations, Pointe Claire, Quebec (Fig. 1). This facility includes a seven-degree-of-freedom crane, with a maximum reach of 7.1 m. The topology of the crane is that of the forwarder machine: its first four joints (RRRP) are hydraulically actuated to position the boom tip in 3D space, with feedback from encoders for R-joints and a magnetic sensor/band for the P-joint. These are followed by two passive rotary joints to support the grapple and the last joint (the rotator), also hydraulically actuated with encoder feedback, allows orienting the grapple with respect to the logs. The crane is integrated on a fixed-base platform, with an attached trailer (5 m long and 2.3 m wide) to hold the logs as they are loaded from the area around the machine.

The perception on the crane relies on the latest AI stereo camera model from Stereolabs, the ZED X. This camera features a wide field of view (120° D), an operational depth range 0.5 m to 20 m and GMSL2 interface. The camera is mounted on the third link of the crane (Fig. 1), close to the elbow joint, to allow both a good overall view around the crane, as well as the view in the vicinity of the grapple when the logs are picked up. The latter will be useful for characterizing the quality of the grasp, in future development.

The log loading experiments were performed on two dozen logs of dry red pine, with lengths between 2.5-2.8 m and diameters between 0.1-0.3 m. The testbed also includes a nearby enclosure which houses the control hardware, with enough room to accommodate two/three people involved in development and experimentation.

B. Control Hardware Architecture

The main elements of the hardware used to control the crane for autonomous operation are depicted in Fig. 5. The host laptop has the following specifications: NVIDIA GeForce RTX 3080 GPU, 16 GB VRAM, Intel i9-11950H CPU and 64 GB RAM. The laptop runs Ubuntu 20.04 with ROS Noetic installation and executes the complete autonomy pipeline. The ZED Box was procured initially to allow to interface the ZED X camera to the host laptop, but in the future, will execute the perception stack of the pipeline to reduce the requirements on the host computer. Currently, we are able to receive RGB-D data from the camera at 9.5 Hz on the ZED Box and 2.8 Hz on the host laptop. The PLC (STW Technic ESX.3CM) is responsible for executing the joint control law based on the reference joint trajectory received from the motion planner. Thus, it interfaces to the crane joint sensors for feedback and sends the computed commands to the electro-hydraulic proportional valves. The control loop is executed at 50 Hz.

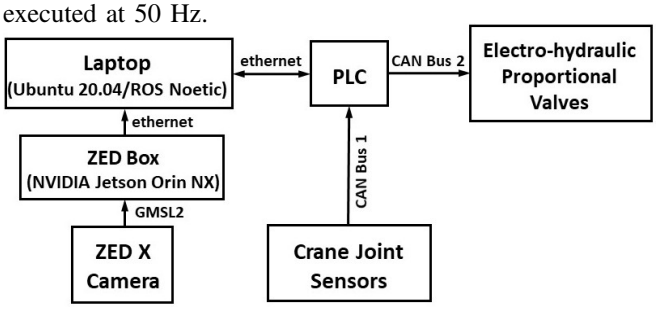


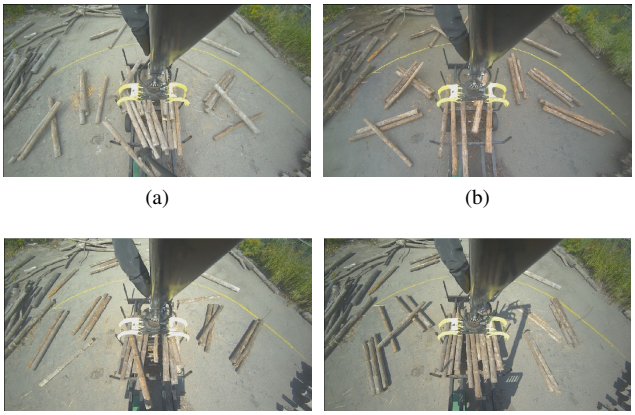
Fig. 5: Control hardware architecture

IV. Experiments On Autonomous Log Loading

A. Description of Experiments

The log loading trials reported here were carried out with the robotic crane testbed over three summer days in partly cloudy, after a rainfall and in sunny weather conditions. A loading cycle was initialized with 12 to 18 logs randomly placed around the trailer, within the workspace of the robot and in the field of view of the camera. For some of the loading cycles, the logs were arranged in single log and small pile (two to four logs) configurations. Example log arrangements at the beginning of four of the loading cycles are illustrated in Fig. 6; an example of the log loading progression over a single cycle is illustrated in Fig. 7. The yellow arc in these figures delineates the reach of the crane.

Autonomous log loading cycle was initiated with the press of a single button on the control user interface on the laptop (e.g., Fig. 7a). The crane then performed a sequence of



(a) (b) (c) (d)

Fig. 6: Examples of log arrangements for four loading cycles (view from ZED X camera): a) cycle 2, b) cycle 6, c) cycle 8, d) cycle 9.

autonomous loading maneuvers (e.g., Fig. 7b and Fig. 7c) until all logs around the trailer were picked up and loaded (e.g., Fig. 7d). Each maneuver started with the crane in home configuration and ended with the crane returning to the same, after the grasped logs were dropped in the trailer, as per maneuver sequence described in section II-D.

In total, ten loading cycles were carried out, which involved a 100 individual loading maneuvers and 153 log counts dropped in the trailer. For each maneuver, we recorded the number of logs grasped, as well as processing times for individual elements of the autonomy pipeline and the total maneuver time. A particular maneuver was considered successful if all parts of the automation were executed and at least one log was delivered to the trailer.

B. Performance Statistics

Performance results for the ten completed log loading cycles are detailed in Table I. Specifically, we present for each cycle: the total number of logs at the beginning of the cycle around the machine, the number of loading maneuvers required to load up all those logs, the number of times a certain number of logs was picked up in that cycle (e.g., 0 picked up logs signifies maneuver failure) and the success rate over the cycle (successful maneuvers/total maneuvers). We observe that out of the 100 maneuvers, there were 42 single log grasps and 45 multiple log grasps; this demonstrates that our grasp planner can produce multiple-log grasps with a single grapple closure. Over the ten cycles, there were 13 failed maneuvers (no logs loaded), for an overall 87% log loading success rate.

We have categorized four types of errors observed over the experimental campaign, summarized in Table II. Note that in some cases, the error resulted in a failed maneuver, and in other cases, the error was observed during the maneuver, but at least one log was loaded into the trailer and hence, was recorded as a successful maneuver. Detailed observations on these errors are narrated below.

Log detection This error occurred when none of the logs around the machine were detected or characterized at the start of the maneuver, usually when only one or two logs remained in the workspace. We typically observed this failure on logs where the shadow of the crane overlapped the log, or in overcast conditions, when the log's colour was washed out, thus blending with the colour of the asphalt background. To correct this, the undetected logs were manually moved to nearby and the log loading cycle was continued.

Final grapple height adjustment Poor performance of this algorithm resulted either in closing the grapple too early, thus missing the log completely, or in a pinch grasp. A light pinch grasp was likely to fail the maneuver because the log was dropped before reaching the trailer. The more solid pinch grasps resulted in successful completion of the maneuver (six successes for this error as indicated in Table II.)

Trajectory tracking Poor trajectory tracking was observed in scenarios where the grapple picked up four large logs because our current joint controller (2) provides no adaptation for the load. Since individual maneuver segments are planned for a fixed time duration, poor trajectory tracking manifests in the crane failing to reach the target position: in the single failure case indicated in Table II for this error, the grapple was commanded to open with the log outside the trailer, resulting in a miss.

Path planning This error represents MoveIt failing to find a solution for a motion plan for a particular segment. In our experiments, this error occurred five times, following a successful grasp. As a result, the system restarted a maneuver from the current failure position with the grasped logs, ultimately depositing the logs successfully into the trailer. In all failure cases, the error occurred due to the inverse kinematics solver unable to find a solution for the path planning target on the first attempt.

C. Timing Statistics

The timing statistics for the main autonomy blocks, specifically, logs segmentation, logs characterization, grasp planning and motion planning are summarized in Table

TABLE I: Log loading performance summary.

Cycle	Total logs	No. maneuvers	Logs per clamp (counts) 0	1	2	≥ 3	Success rate
1	12	10	2	4	4	0	80%
2	14	10	1	5	3	1	90%
3	14	11	2	5	3	1	82%
4	15	15	3	9	3	0	80%
5	16	6	0	1	1	4	100%
6	18	7	0	1	1	5	100%
7	18	14	3	6	3	2	79%
8	13	7	0	3	2	2	100%
9	16	11	2	5	1	3	82%
10	17	9	0	3	4	2	100%
Total	**153**	**100**	**13**	**42**	**25**	**20**	**87%**

TABLE II: Summary of log loading errors.

Error	Counts Failure	Success
Log detection	6	0
Final grapple height adjustment	6	6
Trajectory tracking	1	2
Path planning	0	5

(a) (b) (c) (d)

Fig. 7: Progression of log loading for cycle 5.

III: the stated values represent averages over the successful maneuvers. Also included is the maneuver time as measured from the instance the crane begins to move from the home configuration until it returns and stops at the same. The row annotated 'camera sync delay' indicates the artificially introduced time delay to allow for synchronization of the camera data with the crane location, prior to initiating the perception stack and before the final grapple height adjustment. In the present implementation, we found this delay to be necessary in light of the low rate at which the camera data is received on the host laptop; we expect to eventually reduce this delay with better allocation of computational resources.

The data presented in Table III serve to identify those elements of the autonomy pipeline that incur high computational costs: this in order to guide future development to improve the pipeline. It is noted that the perception and grasp planning stacks are executed only once per loading maneuver, at the very beginning. On the other hand, for the current maneuver design as described in section II-D, the motion planning block is executed six times during the maneuver. Accordingly, the total maneuver time reported in Table III includes 15 seconds for the five intermediate motion plans, as well as the two-second delay for camera synchronization for the final grapple height adjustment.

We can conclude from Table III that the single execution of the grasp planning stack and the motion planning block are currently the more expensive elements of the pipeline. The latter is particularly problematic because of its multiple use during the maneuver. The timing statistics detailed here also provide a basis for comparing the autonomy pipeline delays to typical execution times of a log loading maneuver by a human operator. The time to execute the autonomy pipeline before the loading maneuver begins currently stands, on average, at $\approx$ 9 s. The autonomy requirements during the maneuver execution stand at 17 s, clearly pointing to the need to optimize both the maneuver sequencing and the path planning algorithm. Based on our observations of human executed log loading cycles in mill yards, we estimate times in the range of 20-30 s per loading maneuver: this time includes both the operator's planning intelligence and the maneuver time. Therefore, the present implementation, although of the same order of magnitude, is too slow to compete with a human operator in terms of speed, especially considering that our current autonomy does not include the intelligence for orderly placement of logs in the trailer.

Finally, we note that the total maneuver time for our test-bed (86 s) is significantly higher than that observed for human operators, partly because of the time taken for planning in between waypoints, but mainly because of the limitations of the hydraulic system driving the crane. This results in a slower motion of the crane compared to log loaders in the mill yard, thus contributing to the high average maneuver time reported here.

TABLE III: Average timings for successful maneuvers.

Item	Average time (s)
Camera sync delay	2.0
Logs segmentation	1.12
Logs characterization	0.97
Grasp planning	2.12
Motion planning (IK and MoveIt)	3.0
Pre-maneuver autonomy processing	**9.21**
Camera sync delay	2.0
Motion planning (IK and MoveIt)	3.0 ($\times$5)
Crane motion	69.0
Total maneuver time	**86.0**

V. Conclusions

This paper presents the main components of the autonomy pipeline developed for a log loading operation with a fixed-base crane. Typical of autonomous systems, the pipeline includes a perception stack, a grasp planning stack, as well as, motion planning and control. The autonomy was implemented on a commercial log loader, retrofitted with pre-requisite sensing and control hardware. Notably, perception is enabled with a single crane-mounted stereo camera and an AI-based model pre-trained with images from the testbed and forests. Our grasp planner is agnostic to the number of logs to be grasped, and predicts the best grasp location based on depth image information only. Motion planning and control have been implemented with standard tools and not optimized to date. An extensive experimental campaign to execute ten autonomous log loading cycles of 153 log manipulation counts demonstrated an 87% success rate. Detailed timing results point to the need for further improvements in the maneuver design and the efficiency of the motion planning. Nevertheless, the time requirements to execute the autonomy are in the ballpark of the maneuver times observed of humans operating the log loading crane.

Acknowledgment

This work was supported by the National Sciences and Engineering Research Council (NSERC) Canadian Robotics Network (NCRN), the Mitacs Accelerate Grant for Mr. Ayoub and FPInnovations (through NRCan's TT program.) The log loading testbed was funded, constructed and is maintained by FPInnovations.

REFERENCES

[1] R. Visser and O. F. Obi, "Automation and robotics in forest harvesting operations: Identifying near-term opportunities," *Croatian Journal of Forest Engineering*, vol. 42, no. 1, p. 13 – 24, 2021.

[2] O. Lindroos, O. Mendoza-Trejo, P. La Hera, and D. O. Morales, "Advances in using robots in forestry operations," in *Robotics and automation for improving agriculture*, pp. 233–260, Burleigh Dodds Science Publishing, 2019.

[3] F. Huq and I. Branch, "Skills shortages in canada's forest sector," *Canadian Forest Service*, 2007.

[4] S. Kollarova, *Innovation and Advanced Technology Use in the Canadian Forest Sector*. PhD thesis, Université d'Ottawa/University of Ottawa, 2014.

[5] E. Yousefi, D. P. Losey, and I. Sharf, "Assisting operators of articulated machinery with optimal planning and goal inference," in *International Conference on Robotics and Automation (ICRA)*, pp. 2832–2838, 2022.

[6] S. Westerberg and A. Shiriaev, "Virtual environment-based teleoperation of forestry machines: Designing future interaction methods," *Journal of Human-Robot Interaction*, vol. 2, no. 3, pp. 84–110, 2013.

[7] D. Ortiz Morales, S. Westerberg, P. X. La Hera, U. Mettin, L. Freidovich, and A. S. Shiriaev, "Increasing the level of automation in the forestry logging process with crane trajectory planning and control," *Journal of Field Robotics*, vol. 31, no. 3, pp. 343–363, 2014.

[8] J. Andersson, K. Bodin, D. Lindmark, M. Servin, and E. Wallin, "Reinforcement learning control of a forestry crane manipulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2121–2126, 2021.

[9] H. Gietler, C. Böhm, S. Ainetter, C. Schöffmann, F. Fraundorfer, S. Weiss, and H. Zangl, "Forestry crane automation using learning-based visual grasping point prediction," in *IEEE Sensors Applications Symposium (SAS)*, pp. 1–6, 2022.

[10] S. Weiss, S. Ainetter, F. Arneitz, D. A. Perez, R. Dhakate, F. Fraundorfer, H. Gietler, W. Gubensäk, M. M. D. R. Ferreira, C. Stetco, *et al.*, "Automated log ordering through robotic grasper," in *Austrian Computer Vision and Robotics Workshop*, 2020.

[11] E. Wallin, V. Wiberg, and M. Servin, "Multi-log grasping using reinforcement learning and virtual visual servoing," *arXiv preprint arXiv:2309.02997*, 2023.

[12] J.-M. Fortin, O. Gamache, V. Grondin, F. Pomerleau, and P. Giguère, "Instance segmentation for autonomous log grasping in forestry operations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6064–6071, 2022.

[13] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1290–1299, 2022.

[14] S. Li and H. Lideskog, "Realization of autonomous detection, positioning and angle estimation of harvested logs," *Croatian Journal of Forest Engineering*, 2022.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[16] S. Ainetter, C. Böhm, R. Dhakate, S. Weiss, and F. Fraundorfer, "Depth-aware object segmentation and grasp detection for robotic picking tasks," *arXiv preprint arXiv:2111.11114*, 2021.

[17] S. Ainetter and F. Fraundorfer, "End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from rgb," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13452–13458, 2021.

[18] E. Ayoub, P. Levesque, and I. Sharf, "Grasp planning with cnn for log-loading forestry machine," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11802–11808, 2023.