**Jetson-Enabled Autonomous Vehicle**

**Luis Jibaja1, Alexis Gilmore1, Jose L. Guzman1, Thomas Kwok2, and Hao Jiang2**

**1Cañada College, Redwood City, CA**

**2San Francisco State University, San Francisco, CA**

**Abstract**

Image processing is a popular application within the growing field of AMR (autonomous mobile robots). One significant approach to image processing is HSV segmentation, which extracts significant elements (color, movement, and shapes) of images for analysis. The resultant extracted image data can be utilized for autonomous behavior. The purpose of the present study is to integrate the image processing framework for an autonomous vehicle using NVIDIA’s Jetson TX1 supercomputer. The Jetson TX1 contains 256 CUDA cores which can utilize parallel processing to boost the computational efficiency of data-intensive applications, such as HSV segmentation. It can also be configured for mobile support. To accomplish this goal, we will implement a vehicular line following program that outputs positional error based on deviation from a tracked line. The output error data is fed into a PID control system that manages the vehicle’s motors. The program and interface are developed using ROS, a robotics software framework. With this scheme, we can manipulate the autonomous movement subroutine on a course designed for the vehicle. We can also collect movement and error data to tune feedback parameters. Through this implementation, we present the Jetson TX1 as an accessible controller for an autonomous vehicle, using image processing as a framework.

1. **Introduction**

Image processing applications has continually grown as an interest, creating demand for specialized hardware capable of data-intensive tasks. These tasks require parallel processing to be computationally efficient[[1]](#endnote-1). GPUs (graphics processing unit) provide the hardware for parallel computations to optimize the processing of computer graphics. These processing units are currently very popular in overall parallel data applications, with image processing being one of them. Current trends in image processing programs include a variety of mobile applications, such as vehicles. Within this configuration, programs can be developed to directly manage mobile device peripherals[[2]](#endnote-2). These trends in technology are the main motivating factors for the Jetson line following vehicle.

All internship members were initially unfamiliar with many concepts corresponding to the Jetson vehicle. Therefore, a significant goal was to quickly familiarize with topics such as image processing, control systems, embedded systems, and software development. The laboratory provided a research environment that aided in learning and application. The internship group consistent of one full-time and two half-time members, mentored by a graduate student. The engineering background of all members were diverse, which was handled by appropriately assigning the varied tasks. The overall project timeline was ten weeks, which involved weekly meetings, presentations, and reports.

1. **Jetson TX1 Overview**

NVIDIA Jetson TX1 was used for this project to accommodate the data-intensive software and provide vehicular support. Jetson supercomputer on module was developed to perform the latest consumer and industry-related visual computing applications. Specifically, Jetson TX1 is armed with 256 CUDA (Compute Unified Device Architecture) cores that accelerate the GPU computation processes[[3]](#endnote-3). NVIDIA’s CUDA has developed tools that are compatible with general-purpose programming languages. This compatibility simplifies the incorporation of image processing algorithms with GPU data processes. Furthermore, Jetson TX1 also comes in a small, power-efficient package that is advantageous for mobile applications, such as robots and drones. Jetson TX1 served as the primary system controller for the line following vehicle. It was flashed with Ubuntu OS which supports all relevant software (OpenCV, CUDA, etc.) and hardware (Arduino, motors, camera, etc.).

1. **ROS (Robot Operating System) Overview**

ROS is a robotics software framework that connects low-level devices with high-level, general-purpose programming languages such as C++. ROS includes drivers that support relevant vehicle hardware, such as Jetson TX1 and the Arduino microcontroller. Within the ROS programming workspace, subroutines can be designed to propagate data between supported controllers and peripherals[[4]](#endnote-4). The data manipulation within the ROS subroutines allows the implementation of image processing and control system algorithms that manage the vehicle’s movement.

ROS has a networked communication system for instructions and data retrieval. Device actions, such as rotating a motor, have their own specific high-level subroutines that contain the files necessary to access lower-level control[[5]](#endnote-5). Instructions made for the overall movement routine calls upon a networked collection of device action subroutines to perform a command. Device data, such as motor speed, can be accessed by calling upon the device action subroutines.

**4. PID Control System Overview**

Since we must apply a continuous correction when the robot navigates a path, we implement an error feedback system with the Jet-vehicle. We found that a PID system can give the robot the ability to regulates its driving process autonomously due to its error control. PID (Proportional Integral Derivative) is a control loop feedback mechanism that quickly minimizes an error value in a control system[[6]](#endnote-6). PID continuously calculates the error value as the difference between the desired setpoint and a measured process variable with the purpose to apply a correction using the proportional (, integral that stands for the past error values, and derivative ; this derivative is also called the “anticipatory control” because one estimate the future trend of the error value[[7]](#endnote-7). The system is based on the total sum of the current, derivative and the integral of the error to automatically applies an accurate and responsive correction in a control function or in this project when the Jet-vehicle drive following a path.



Fig 1: Block diagram of a PID controller in a feedback loop

Fig 1 shows how to generate a full connected PID system. The block diagram explains how the error value is calculated constantly as the difference between the desired setpoint ( and the measured process variable (.

 (1)

At the same time can apply a correction based on the proportional, integral, and derivative terms. The whole system attempts to minimize the error over time by adjustment of a control variable. The overall control value can be simplified by the summation of the main three components of the PID. The mathematical form can be expressed as

 (2)

where , and are non-negative real numbers that should be balanced by the loop tuning. For the purposes of this research “Ziegler-Nichols” techniques will be used for tuning the PID parameters. “Ziegler-Nichols” propose to set the and to zero, and the increased from zero until the control loop reaches a stable and consistent control output[[8]](#endnote-8). Tuning a PID controller takes some patience, but the result is a highly efficient control system.

**5. HSV Segmentation Overview**

With the goal to enable an autonomous behavior for the Jet-vehicle, our model implements an image processing technique such as HSV segmentation, with the purpose that the robot can be able to detect a color line on the ground through a usb-camera. HSV is a variant of the RGB color model which is an additive color model in which red, green and blue light are added together in various ways to reproduce a broad array of colors[[9]](#endnote-9). HSV is different than the RGB model since this technique implement a colored system that describes a hue shift, saturation and value of each color at the spectrum. In computer graphics, HSV model provides the ability to combine different amounts of red, green or blue lights to create complex shades; due to this feature we can implement a fully segmentation process to extract the most significant characteristics of an object in an image.

In our model, we implement an image segmentation process where the robot first receives image frames as input through the camera and process these images as HSV matrices for each pixel. HSV-matrices have three numbers that stand for the hue(color), saturation(whiteness) and value (lightness of the pixel); As each image is already transformed into HSV-matrices we perform a color-base segmentation where the image input is partitioning into multiple segments to assign a label to each pixel which helps to differentiate each characteristic in the picture of the color line while the robot navigate a path. To accomplish this goal, we use OpenCV which is an opensource computer vision library that provides pre-build vision operations that can interact with ROS.

**6. PID-Jet Control System (PID-JCS)**

To prepare an ideal, lightweight control system for an accurate and responsive autonomous robot; a full error feedback system was implemented on the Jet-vehicle. PID-Jet Control System is a fully connected structure that implements HSV segmentation for image processing with the objective to receive input image frames for map localization and provides a continuous error feedback for the PID algorithm which provides a smooth drive down the length of a traced line. For this study, a PID-JCS architecture was implemented on the Jetson TX1 with the objective to make the Jet-vehicle drive through a path following a line autonomously.

As part of the PID-JCS architecture has the constraint that a Logitech webcam should be on the front of the Jet-vehicle’s chassis pointing downwards to the floor. The purpose is to receive constant image frame input from a color line which the robot should be able to recognize and track the image object. To accomplish this goal HSV segmentation is used to detect the pixels in the color line. The segmentation process extracts certain components from the color line image and the algorithm creates an HSV matrix to compute the image moment that needs to obtain a final mask matrix; this feature helps to determine the limits of the line and create a clear picture of the segmented object[[10]](#endnote-10). As the color line has a clean segmentation, the deviation of the robot following the line can be calculated using the image moment which determines the “center of mass” of the color line[[11]](#endnote-11). As the robot deviates from the “center of mass” of the color line, the HSV algorithm uses this deviation as continuous error feedback which is used as input for the PID algorithm.



Fig 2: Line segmentation with the center of mass recognition

The PID algorithm was feed by a constant error feedback which is provided by the HSV system. The purpose of the PID structure is proportion continuous drive correction as the Jet-Vehicle follow the color line. The PID system will be set by a function called that retrieve the error feedback and set the the current error as the proportional, the integral as the summation of the current error, and the derivative as the difference between the current and previous error. The goal is established a constant correction on the angular velocity as the control value which should maintain a balanced driving meanwhile the robot follows the colored line in a constant linear velocity .

 (3)

The gain parameters should be adjusted in the PID algorithm using the “Ziegler-Nichols” technique already discuss; as the integral value can get very large, so often good results can be achieved with very small values of even zero for the term. As the current algorithm suggest adjusting values are critical because of error messages receive from the HSV algorithm.

Fig 3: Algorithm implementation of a PID model on a Jet-Control System

The algorithm in Fig 3 suggests a whole implementation of PID model with gain values as a proportional scalar with the goal to achieve a continuous correction on the angular velocity .

**7. Results**

After build a complete embed system with the Jetson TX1 and the robotic hardware that can have a full interaction with ROS. We were able to implement a functional image recognition model that create a clear picture of segmented objects and determine the limits of a colored line. As the PID-JCS requires an error feed for the driving correction, the HSV-segmentation algorithm accomplish the goal providing continuous error data that represent the vehicle displacement from the edges of the line. To provide a stable driving, we minimize our error data by tuning the PID parameters in our algorithm using the “Ziegler-Nichols” technique. For the experiment test, the main metric indicates the error minimization by comparing how the position displacement evolves over time where

|  |
| --- |
| Gain Parameter Analysis |
|  | 0.097 |
|  | 0.0 |
|  | 0.0 |
| Critical Value | 200 |

Table 1: Initial baseline for tuning

Table 1 show how we apply “Ziegler-Nichols” technique, by setting the , and parameters by zero and term by a small number. The purpose is increased the term until the control loop reaches a stable and minimum error output. Also, the integral integral parameter was set as zero for the whole process to eliminate the continuous accumulation of error.

****

Fig 4: Positional Displacement[mm] vs Time[s], incrementation

By tuning the term with its initial value, Fig 4 shows how this term produce that the graph diverge fast until reaches the critical value what really means that the robot deviates 200 millimeters out from the segmented region on the blue-line. When the robot reaches this displacement means that the robot cannot recognize any colored line anymore. At the same time, the graph shows a comparison between the initial value for the and how the increment of this value, minimize the position displacement as the time evolves. Also, using a polar coordinate system, positive values represent the vehicle’s displacement from the right edge of the line, and negative values when the vehicle deviates from the left edge of the line. As result we found the minimum position displacement is when is equal to 0.01, if the value for the parameter increase then the error rise again.

|  |
| --- |
| Gain Parameter , Analysis |
|  | 0.01 |
|  | 0.0 |
|  | 0.01 |

Table 2: Initial baseline, tuning

Applying the same technique, the goal is increase parameter to find the optimal minimum positional displacement while we maintain the and .values. Table 2 shows how the is set as the term since this parameter has already its optimized value; the reason is that the tunning of the parameter can keep the same minimum error as the term or better.



Fig 5: Positional Displacement[mm] vs Time[s], tuning

At Fig 5 shows the results from tuning the term, as result the position displacement gets to the minimum values when is set equal to 0.3. If the parameter is incremented after this value, the position displacement values start to diverge farther from zero. As a final result, the minimum position displacement can be achieved by the optimized PID parameters on table 3 which produce a stable vehicle’s movement.

|  |
| --- |
| Final Gain Parameters  |
|  | 0.01 |
|  | 0.0 |
|  | 0.3 |

Table 3: Final PID parameters for stable vehicle tracking

**8. Conclusion**

Due to the accuracy and algorithm performance, Jetson TX1 was a perfect choice for its capabilities such as parallel computation for image processing. Also, since the PID system was simple to implement, Jetson TX1 could handle a full control system feedback to build an autonomous driving vehicle. As result, we could tune the final optimized parameters for the PID-Jet control system, so the robot was able to get a full autonomous behavior. Where the Jet-Vehicle can use the HSV segmentation to identify a color line, and the tuned PID parameters actively stabilized vehicle’s movement along the line. Due to the minimum position displacement from the colored line the robot can execute a continuous correction and producing a perfect navigation while follow the color line.

**Acknowledgment**This project is supported by the US Department of Education through the Minority Science and Engineering Improvement Program (MSEIP, Award No. ); and through the Hispanic-Serving Institution Science, Technology, Engineering, and Mathematics (HSI STEM) Program, Award No.

**Bibliography:**

1. Zhang, Nan, Yun-shan Chen, and Jian-li Wang. "Image parallel processing based on GPU." In *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, vol. 3, pp. 367-370. IEEE, 2010. [↑](#endnote-ref-1)
2. Tang, Jie, Yong Ren, and Shaoshan Liu. "Real-Time Robot Localization, Vision, and Speech Recognition on Nvidia Jetson TX1." *arXiv preprint arXiv:1705.10945* (2017). [↑](#endnote-ref-2)
3. Otterness, Nathan, Ming Yang, Sarah Rust, Eunbyung Park, James H. Anderson, F. Donelson Smith, Alex Berg, and Shige Wang. "An evaluation of the NVIDIA TX1 for supporting real-time computer-vision workloads." In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017 IEEE*, pp. 353-364. IEEE, 2017. [↑](#endnote-ref-3)
4. Quigley, Morgan, Brian Gerkey, and William D. Smart. *Programming Robots with ROS: a practical introduction to the Robot Operating System*. " O'Reilly Media, Inc.", 2015. [↑](#endnote-ref-4)
5. Quigley, Morgan, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. "ROS: an open-source Robot Operating System." In *ICRA workshop on open source software*, vol. 3, no. 3.2, p. 5. 2009. [↑](#endnote-ref-5)
6. Ang, Kiam Heong, Gregory Chong, and Yun Li. "PID control system analysis, design, and technology." *IEEE transactions on control systems technology* 13, no. 4 (2005): 559-576. [↑](#endnote-ref-6)
7. Zhao, Pan, Jiajia Chen, Yan Song, Xiang Tao, Tiejuan Xu, and Tao Mei. "Design of a control system for an autonomous vehicle based on adaptive-PID." *International Journal of Advanced Robotic Systems* 9, no. 2 (2012): 44. [↑](#endnote-ref-7)
8. Meshram, P. M., and Rohit G. Kanojiya. "Tuning of PID controller using Ziegler-Nichols method for speed control of DC motor." In *Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on*, pp. 117-122. IEEE, 2012. [↑](#endnote-ref-8)
9. Sural, Shamik, Gang Qian, and Sakti Pramanik. "Segmentation and histogram generation using the HSV color space for image retrieval." In *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 2, pp. II-II. IEEE, 2002. [↑](#endnote-ref-9)
10. Liao, Simon Xinmeng, and Miroslaw Pawlak. "On image analysis by moments." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18, no. 3 (1996): 254-266. [↑](#endnote-ref-10)
11. Meyer, Fernand. "Color image segmentation." In *Image Processing and its Applications, 1992., International Conference on*, pp. 303-306. IET, 1992. [↑](#endnote-ref-11)