



Pioneer Autônomo com Carmen-LCAD

Documentação

12/12/2019

Anderson Mozart, Gabriel Nunes, Genilson Cruz, Leandro de Lima, Marcos Piumbini,
Thiago Cavalcante

Robótica Probabilística

Profª Ph.D Claudine Badue

UFES

2019

Visão geral

O desafio principal do projeto é realizar a integração do Pioneer com o framework Carmen - LCAD e fazer o robô Pioneer trafegar autônomo indoor com o objetivo final de, se possível, trafegar até a cantina. Um dos desejos é verificar a possibilidade do robô trafegar em caminhos diferentes do que a IARA trafega, como ambientes indoor e caminhos como passarelas e calçadas. Para que a integração seja completa iremos simular o Pioneer, que é um robô diferencial, em um robô ackermann, ou seja, iremos colocar a função que se locomove feito um robô ackermann, com as rodas dianteiras que comandam a direção e as rodas traseiras que fornecem a tração.

Sumário

Visão geral	1
Sumário	1
Especificações do projeto	2
Etapas do projeto	3
1 - Entendendo o Pioneer	3
2 - Entendendo a Sensorbox	4
3 - Integrando a Sensorbox ao Pioneer	4
4 - Integrando o driver do Pioneer no Github do Carmen-LCAD [3]	5
5 - Transformar o movimento do Pioneer em Ackermann	9
6 - Ajustar parâmetros do Pioneer no carmen-pioneer-sensorbox.ini	9
7 - Controlar o Pioneer através do Joystick	10
8 - Criar process (navigate, log, mapper) para o Pioneer com a Sensorbox	11
9 - Gravar um Log utilizando o Pioneer e a Sensorbox	11
10 - Função de calibração da odometria sem GPS	12
11 - Utilizar o log para criar um mapa	12
12 - Ajustar parâmetros de PID	12
13- Fazer o pioneer se localizar e se mover autônomo em um RDDF específico	13
Discussões e Conclusões do projeto	14
Melhorias para serem realizadas no CARMEN-LCAD para facilitar integração	14
Dificuldades que tivemos	14
Aprendizado com o mundo real	14

Especificações do projeto

- Fazer o Pioneer, integrado à Sensorbox, se mover através da função de Ackermann;
- Ajustar parâmetros de todas as medições do robô para o pioneer no `carmen-pioneer-sensorbox.ini` ;
- Controlar o Pioneer através do Joystick;
- Criar process (navigate, log, mapper) para o Pioneer com a Sensorbox;
- Gravar um Log utilizando o Pioneer e a Sensorbox;
- Criar o mapa usando as funções de SLAM;
- Ajustar parâmetros do PID diferencial para que o robô trafegue de forma suave sem sair do RDDF definido;
- Trafegar autônomo utilizando as funções de robótica probabilística implementadas no CARMEN-LCAD;
- Todos os códigos-fonte devem ficar disponibilizados no GitHub do CARMEN-LCAD [3].

Etapas do projeto

1 - Entendendo o Pioneer

O Pioneer 3-DX é um pequeno robô de tração diferencial leve de duas rodas e dois motores, ideal para uso em laboratório ou em sala de aula [1]. O Pioneer P3-DX é uma base robótica com rodas projetada para navegação autônoma. É a plataforma de pesquisa móvel mais popular do mundo [2]. As especificações do Pioneer [2] são:

CARACTERÍSTICAS

Equipado com chassi robusto e plataforma de montagem com capacidade para 17 kg.

O robô possui três baterias de 7 amperes por hora, 12 volts de corrente contínua (VDC), seladas de chumbo/ácido (total de 252 watts-hora), acessíveis através de uma porta na parte traseira.

ALTURA

24 cm | 9.4 in

COMPRIMENTO

45 cm | 17.7 in

LARGURA

38 cm | 15 in

PESO

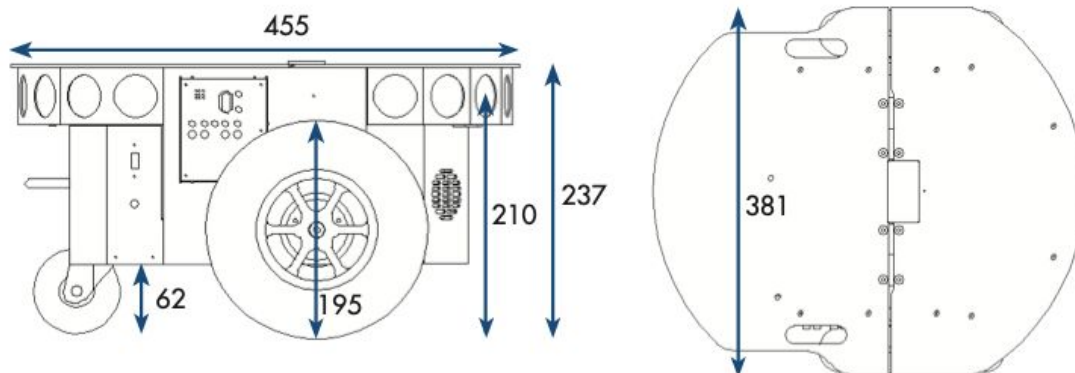
9 kg | 19.8 lb

VELOCIDADE

4.32 km/h | 2.7 mph



Dimensions (mm)



2 - Entendendo a Sensorbox

A Sensorbox é um conjunto de sensores em uma caixa de acrílico de aproximadamente 5 Kg que possui: uma unidade de LiDAR de 32 canais modelo Velodyne HDL-32E [6]; uma Câmera; uma IMU (Unidade de Medição Inercial); duas unidades de GPS Reach; um Roteador e um Raspberry Pi. Para se conectar na Sensorbox é utilizado um cabo RJ-45 e um cabo de alimentação.

As especificações dos IPs dos sensores da Sensorbox são:

Velodyne: 192.168.1.202 ; RaspberryPi: 192.168.1.15 ; Roteador Multilaser: 192.168.2.1 ; GPS Reach 1: 192.168.1.16:81 ; GPS Reach 2: 192.168.1.16:82.

3 - Integrando a Sensorbox ao Pioneer

A Sensorbox foi presa na parte superior do Pioneer através do uso de adesivos de velcro em uma posição específica. A posição definida foi escolhida de forma que o centro da Sensorbox ficasse o mais próximo possível do centro do eixo das rodas traseiras do Pioneer e o centro de gravidade do conjunto ficasse em uma região segura contra tombamento, entre a roda guia e o eixo das rodas traseiras.

O controle do Pioneer e a leitura dos dados da Sensorbox é feita por um notebook apoiado em cima da Pioneer e preso à Sensorbox com o uso de abraçadeiras de Nylon. A comunicação da Sensorbox com o notebook é feita através de um cabo Ethernet e a comunicação do notebook com o Pioneer é feita através de um cabo Serial RS232 para USB .

O Pioneer possui um conjunto de baterias internas que, além de alimentar o próprio robô, fornecem energia para o funcionamento da Sensorbox. Já o notebook é alimentado por bateria própria.

Para comunicar o computador com a Sensorbox é adicionada uma rede manual, renomeado para Sensorbox, e adicionado manualmente os seguintes IPs:

192.168.1.1 192.168.0.1 Ambos com netmask 24



4 - Integrando o driver do Pioneer no Github do Carmen-LCAD [3]

O framework Carmen utiliza IPC - Inter Process Communication para que os módulos se comuniquem entre si.

O IPC fornece passagem flexível e eficiente de mensagens entre processos. Ele pode enviar e receber de forma transparente estruturas de dados complexas, incluindo listas e matrizes de tamanho variável, usando os paradigmas anônimos de "publicar / assinar" e "cliente / servidor". Uma grande variedade de idiomas e sistemas operacionais são suportados [5].

Na versão do framework carmen 0.7.4 disponível em [4] há o driver para o Pioneer 3-DX através da comunicação da porta serial, dentro da pasta /src/base. Utilizamos esse driver e adaptamos para o framework Carmen-LCAD modificando e adicionando as seguintes linhas no arquivo **src/base/base_main.c** para que o robô realize as funções de movimento e publique a odometria:

Adição das linhas com include do código na linha 31:

```
#include <carmen/base_interface.h>

#include <carmen/base_ackerman_interface.h>

#include <carmen/robot_ackerman_interface.h>
```

Adição da variável global para transformá-lo em ackermann na linha 65:

```
static double distance_between_front_and_rear_axles;
```

Para preencher essa variável, carregá-la do arquivo de parâmetros carmen-pioneer-sensorbox.ini, na linha 277 :

```
{"robot", "distance_between_front_and_rear_axles", CARMEN_PARAM_DOUBLE,
    &(distance_between_front_and_rear_axles), 1, NULL}};
```

Adicionada as duas funções na linha 386, uma que irá se inscrever para receber o comando de movimento ackermann e outra para publicar a mensagem de odometria.

O handler utilizado tem o nome de **base_ackerman_subscribe_motion_command_handler**, ele foi criado com base no método que já existia no código, com o nome **velocity_handler**, mas foram feitas algumas modificações para funcionar com o framework Carmen-LCAD.

Foi criado o método **publish_odometry_message** no **base_main.c** (necessário dar include no **<carmen/robot_ackerman_interface.h>**) e dar define nas mensagens tipo **CARMEN_ROBOT_ACKERMANN_VELOCITY_NAME**.

```
static void
base_ackerman_subscribe_motion_command_handler(carmen_base_ackerman_motion_command_mes
sage *motion_command_message)
{
    carmen_ackerman_motion_command_p vel;

    vel = motion_command_message[0].motion_command;

    // double L = distance_between_front_and_rear_axles; // Distancia entre eixos.

    double delta_theta;

    int base_err;

    if(vel->v == 0 && vel->phi == 0)
    {
        if (moving)
        {
            do
            {
                base_err = carmen_base_direct_set_deceleration(deceleration);

                if (base_err < 0)

                initialize_robot();

            }

            while (base_err < 0);

            moving = 0;
        }

        carmen_warn("S");

    }

    else if (!moving)
    {

        moving = 1;

        current_acceleration = robot_config.acceleration;
```

```

        do
        {
            base_err = carmen_base_direct_set_acceleration(current_acceleration);

            if (base_err < 0)

                initialize_robot();
        }

        while (base_err < 0);

        carmen_warn("V");

    }

    if (odometry_inverted)

        vel->v *= -1;

        vel->phi *= -1; // trecho acrescentado por Gabriel Hendrix

    if (!use_hardware_integrator)

    {

        vel->v /= relative_wheelsize;

        vel->phi /= relative_wheelsize;

        vel->phi /= relative_wheelbase;

    }

    do

    {

        delta_theta = (vel->v / distance_between_front_and_rear_axles) *
tan(vel->phi);

        base_err = carmen_base_direct_set_velocity(vel->v, delta_theta);

        last_motion_command = carmen_get_time();

        if (base_err < 0)

            initialize_robot();

    }

    while (base_err < 0);
}

```



```

void
publish_odometry_message(carmen_base_odometry_message current_odometry)
{
    IPC_RETURN_TYPE err;

    carmen_robot_ackerman_velocity_message odometry_message;

    odometry_message.v = current_odometry.tv;

    //odometry_message.phi = (odometry_message.v/ distance_between_front_and_rear_axles)
    * tan(current_odometry.rv);

    if(odometry_message.v == 0.0)

        odometry_message.phi = 0.0;

    else

        odometry_message.phi =
atan((distance_between_front_and_rear_axles/odometry_message.v) *
current_odometry.rv);

    odometry_message.host = current_odometry.host;

    odometry_message.timestamp = current_odometry.timestamp;

    err = IPC_publishData(CARMEN_ROBOT_ACKERMAN_VELOCITY_NAME, &odometry_message);

    carmen_test_ipc_exit(err, "Could not publish", CARMEN_ROBOT_ACKERMAN_VELOCITY_FMT);
}

```

Na pasta "base", o código que controla o pioneer é o `base_main.c`, no método `carmen_base_initialize_ipc` foi inserido o subscribe para receber a mensagem que o controle está publicando, foi utilizado o método `carmen_base_ackerman_subscribe_motion_command` para isso (para utilizar esse método é necessário dar o include no `carmen/base_ackerman_interface.h`).

Foi adicionado na inicialização do IPC para definir a mensagem de velocidade linha 571:

```

err = IPC_defineMsg(CARMEN_ROBOT_ACKERMAN_VELOCITY_NAME, IPC_VARIABLE_LENGTH,

    CARMEN_ROBOT_ACKERMAN_VELOCITY_FMT);

    carmen_test_ipc_exit(err, "Could not define", CARMEN_ROBOT_ACKERMAN_VELOCITY_NAME);

```

E na linha 592 a inscrição para receber o comando de movimento:

```
carmen_base_ackerman_subscribe_motion_command(NULL,
        (carmen_handler_t) base_ackerman_subscribe_motion_command_handler,
        CARMEN_SUBSCRIBE_LATEST);
```

Na função `carmen_base_run`, na linha 743, foi adicionado a publicação da odometria:

```
publish_odometry_message(odometry);
```

5 - Transformar o movimento do Pioneer em Ackermann

No `base_ackerman_subscribe_motion_command_handler` foi inserido o cálculo para se obter a rotação de um veículo ackerman na linha 440, e então o resultado desse cálculo é enviado pelo método `carmen_base_direct_set_velocity` na linha 441 junto com a velocidade original.

```
delta_theta = (vel->v / distance_between_front_and_rear_axles) * tan(vel->phi);
        base_err = carmen_base_direct_set_velocity(vel->v, delta_theta);
```


6 - Ajustar parâmetros do Pioneer no `carmen-pioneer-sensorbox.ini`

Foi copiado o `/src/carmen-ford-escape-sensorbox.ini` para outro arquivo com o nome `carmen-pioneer-sensorbox.ini`. Nesse novo arquivo, foram inseridos os seguintes parâmetros:

```
base_type
base_dev
base_model
base_motion_timeout
base_use_hardware_integrator
base_relative_wheelsize
base_relative_wheelbase
robot_odometry_inverted
```

Foram modificados os seguintes parâmetros:

```
robot_length
robot_width
robot_vertical_displacement_from_center
```



```
robot_wheel_radius
robot_acceleration
robot_deceleration
robot_distance_between_front_and_rear_axles
```

Criada pasta `pioneer_p3dx` em bin com o arquivo de colisão do robô. Trocado o parâmetro `robot_collision_file` e foi ligada a opção `show_collision_range` em `carmen-ford-escape-sensorbox.ini`, para representar o robô como um círculo no navigator_gui.

Ao executar o `process-volta_no_ct9_log_sensorbox.ini`, o viewer 3d informava erro no arquivo de parâmetros `carmen-pioneer-sensorbox.ini`. Foram verificados e incluídas as seguintes seções no arquivo (retirado do arquivo `carmen-ford-escape.ini`):

```
-#----- Bumblebee Basic 7 REALSENSE -----
#----- Stereo Velodyne 7 (Realsense)-----
#----- Camera 7 - RealSense -----
```

E foram alterados alguns parâmetros do Robot Ackermann para as medidas feitas para o pioneer. Os parâmetros alterados foram:

```
robot_max_velocity
robot_distance_between_rear_wheels
robot_distance_between_rear_car_and_rear_wheels
robot_distance_between_front_car_and_front_wheels
robot_distance_between_rearview
robot_turning_radius
```

Também foram alterados parâmetros da sensor board:

```
sensor_board_1_x
sensor_board_1_y
sensor_board_1_z
sensor_board_1_yaw
sensor_board_1_pitch
sensor_board_1_roll
```

Segue Lista completa com as alterações de parâmetros:

SEÇÃO	PARÂMETRO	FORD ESCAPE	PIONEER	COMENTÁRIOS
Localize Ackerman parameters	localize_v_uncertainty_at_zero_v	N/A	0.0	
Localize Ackerman parameters	localize_remission_variance_multiplier	N/A	2.0	
Map Server	map_server_initial_waiting_time	3.0	0.0	
Map Server	map_server_map_grid_res	0.2	0.05	
Map Server	map_server_map_width	210.0	15.0	
Map Server	map_server_map_height	210.0	15.0	
Map Server	map_server_time_interval_for_map_change	0.3	0.1	
Mapper Parameters	mapper_map_grid_res	0.2	0.05	
Mapper Parameters	mapper_map_width	210.0	15.0	
Mapper Parameters	mapper_map_height	210.0	15.0	
Mapper Parameters	mapper_mapping_mode_off_velodyne_range_max	70.0	10.0	
Mapper Parameters	mapper_mapping_mode_on_velodyne_range_max	70.0	10.0	
Model Predictive Planner parameters	model_predictive_planner_obstacles_safe_distance	1.6	0.2	
Model Predictive Planner parameters	model_predictive_planner_obstacles_safe_length_distance	5.63	0.47	

Model Predictive Planner parameters	model_predictive_planner_max_square_distance_to_lane	2.0	0.5	
Obstacle Avoider parameters	obstacle_avoider_obstacles_safe_distance	1.0	0.1	
Behavior Selector parameters	behavior_selector_central_lane_obstacles_safe_distance	1.6	0.6	
Behavior Selector parameters	behavior_selector_main_central_lane_obstacles_safe_distance	1.1	0.5	
Behavior Selector parameters	behavior_selector_lateral_lane_obstacles_safe_distance	1.1	0.5	
Behavior Selector parameters	behavior_selector_lateral_lane_displacement	2.2	1.0	
Behavior Selector parameters	behavior_selector_distance_between_waypoints	13.0	1.5	
Behavior Selector parameters	behavior_selector_change_goal_distance	13.0	1.5	
Robot parameters	base_type	N/A	pioneer	
Robot parameters	base_dev	N/A	/dev/ttyUSB0	
Robot parameters	base_model	N/A	p2d8+	
Robot parameters	base_motion_timeout	N/A	1.0	
Robot parameters	base_use_hardware_integrator	N/A	off	
Robot	base_relative_wheelsize	N/A	1.0	

parameters				
Robot parameters	base_relative_wheelbase	N/A	1.0	
Robot parameters	robot_length	4.425	0.445	
Robot parameters	robot_width	1.806	0.393	
Robot parameters	robot_vertical_displacement_from_center	-1.375	0.0	
Robot parameters	robot_wheel_radius	0.28	0.095	
Robot parameters	robot_acceleration	2.00	0.18	
Robot parameters	robot_deceleration	0.30	0.90	
Robot parameters	robot_odometry_inverted	N/A	on	
Robot parameters	robot_polygon_file	ford_escape / ford_escape_poly.txt	pioneer_p3dx / pioneer_p3dx_poly.txt	
Robot parameters	robot_collision_file	ford_escape / ford_escape_collision.txt	pioneer_p3dx / pioneer_p3dx_collision.txt	
Robot Ackermann parameters	robot_max_steering_angle	0.5537	0.8	
Robot Ackermann parameters	robot_max_velocity	8.33	0.4	alterados parâmetros do Robot Ackermann para as medidas feitas para o pioneer
Robot Ackermann parameters	robot_distance_between_front_and_rear_axles	2.625	0.4	alterados parâmetros do Robot Ackermann para as medidas

				feitas para o pioneer
Robot Ackermann parameters	robot_distance_between_rear_wheels	1.535	0.33	alterados parâmetros do Robot Ackermann para as medidas feitas para o pioneer
Robot Ackermann parameters	robot_distance_between_rear_car_and_rear_wheels	0.96	0.19	alterados parâmetros do Robot Ackermann para as medidas feitas para o pioneer
Robot Ackermann parameters	robot_distance_between_front_car_and_front_wheels	0.85	0.09	alterados parâmetros do Robot Ackermann para as medidas feitas para o pioneer
Robot Ackermann parameters	robot_distance_between_rearview	2.065	0.4	alterados parâmetros do Robot Ackermann para as medidas feitas para o pioneer
Robot Ackermann parameters	robot_turning_radius	5.6	3.0	alterados parâmetros do Robot Ackermann para as medidas feitas para o pioneer
Robot Ackermann parameters	robot_maximum_steering_command_rate	0.335	1.0	
Robot Ackermann parameters	robot_understeer_coeficient	0.0015	1.0	
Robot Ackermann	robot_maximum_speed_forward	46.0	1.5	

parameters				
Robot Ackermann parameters	robot_maximum_speed_reverse	20.0	1.0	
Robot Ackermann parameters	robot_v_multiplier	1.015018	1.0	Ajuste da odometria do carro
Robot Ackermann parameters	robot_phi_multiplier	1.096677	1.0	Ajuste da odometria do carro
Robot Ackermann parameters	robot_PID_steering_kp	689.4	1000.0	Ajuste da odometria do carro
Robot Ackermann parameters	robot_PID_steering_ki	2008.7	2000.0	Ajuste da odometria do carro
Robot Ackermann parameters	robot_PID_steering_kd	30.8	90.0	Ajuste da odometria do carro
Grid Mapping Map parameters	grid_mapping_map_grid_res	0.2	0.05	
Grid Mapping Map parameters	grid_mapping_map_width	210	15	
Grid Mapping Map parameters	grid_mapping_map_height	210	15	
Sensor Board 1 parameters	sensor_board_1_x	3.40	0.025	
Sensor Board 1 parameters	sensor_board_1_z	0.737	0.14	

Velodyne Variable Scan Message parameters				Criada seção no arquivo pioneer.ini mas não achei referência a alguns do parâmetros. Devemos documentar ou manter ? Eles foram usados apenas para teste ?
Velodyne parameters	velodyne_z	0.090805	0.35	
Velodyne parameters	velodyne_pitch	-0.0227	0.0	
Velodyne parameters	velodyne_yaw	-0.01	0.0	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_width	N/A	640	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_height	N/A	480	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_fx	N/A	0.605578	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_fy	N/A	0.807437	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_cu	N/A	0.498175	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_cv	N/A	0.512542	
Bumblebee	bumblebee_basic7_baseline	N/A	0.05	

Basic 7 REALSENSE				
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_pixel_size	N/A	0.00000375	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_is_legacy	N/A	off	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_is_rectified	N/A	on	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_model	N/A	xb3	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_fov	N/A	0.6	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_tlight_roi_x	N/A	0	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_tlight_roi_y	N/A	0	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_tlight_roi_w	N/A	640	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_tlight_roi_h	N/A	480	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_tlight_focal_dist	N/A	1500.0	
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_tlight_dist_correction	N/A	6.0	
Bumblebee Basic 7	bumblebee_basic7_stereo_stride_x	N/A	10.0	

REALSENSE				
Bumblebee Basic 7 REALSENSE	bumblebee_basic7_stereo_stride_y	N/A	10.0	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_num_points_cloud	N/A	1	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_flipped off	N/A	off	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_horizontal_resolution	N/A	640	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_vertical_resolution	N/A	480	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_range_max	N/A	2.0	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_horizontal_roi_ini	N/A	0	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_horizontal_roi_end	N/A	640	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_vertical_roi_ini	N/A	0	
Stereo Velodyne 7 (Realsense)	stereo_velodyne7_vertical_roi_end	N/A	480	
Camera 7 – RealSense	camera7_x	N/A	3.0	
Camera 7 – RealSense	camera7_y	N/A	0.0	

Camera 7 – RealSense	camera7_z	N/A	-0.9	
Camera 7 – RealSense	camera7_roll	N/A	0	
Camera 7 – RealSense	camera7_pitch	N/A	2.09	
Camera 7 – RealSense	camera7_yaw	N/A	3.14	
Viewer_3D parameters	carmodel_x	1.2	0.09	Parâmetros alterados para corrigir erro ao rodar módulo.
Viewer_3D parameters	carmodel_z	0.28	-0.28	Parâmetros alterados para corrigir erro ao rodar módulo.
Viewer_3D parameters	carmodel_size_x	4.437	0.43	Parâmetros alterados para corrigir erro ao rodar módulo.
Viewer_3D parameters	carmodel_size_y	1.806	0.38	Parâmetros alterados para corrigir erro ao rodar módulo.
Viewer_3D parameters	carmodel_size_z	1.725	0.24	Parâmetros alterados para corrigir erro ao rodar módulo.
Laser position and orientation in relation to car	xsens_magnetic_declination	N/A	-23.457849	
Logger parameters	logger_imu_pi	on	off	
Logger parameters	logger_xsens_mtig	off	on	



Logger parameters	logger_ford_escape_status	on	off	
----------------------	---------------------------	----	-----	--

7 - Controlar o Pioneer através do Joystick

Modificamos o código `~/carmen_lcad/src/joystick/wingman_control.c` para que o método `send_base_velocity_command` fosse capaz de enviar uma mensagem do tipo `carmen_base_ackerman_velocity_message` e realizar a publicação dessa mensagem. Para utilizar esse método, é necessário dar o include do `carmen/obstacle_avoider_interface.h`. Também foi modificado o método `define_messages` para que seja definido as mensagens do tipo `CARMEN_BASE_ACKERMAN_MOTION_COMMAND_NAME`. Também foram comentadas algumas seções do método `main` para que a opção "throttle_mode" não fosse utilizada, já que não será necessária para a nossa aplicação e a ativação do modo acelerador estava causando bugs no funcionamento.



8 - Criar process (navigate, log, mapper) para o Pioneer com a Sensorbox

Foi criado o `process-pioneer_sensorbox_playback_viewer_3D.ini` e ao executá-lo, foi necessário incluir no `carmen-pioneer-sensorbox.ini` os seguintes parâmetros:

```
localize_v_uncertainty_at_zero_v
localize_remission_variance_multiplier
```

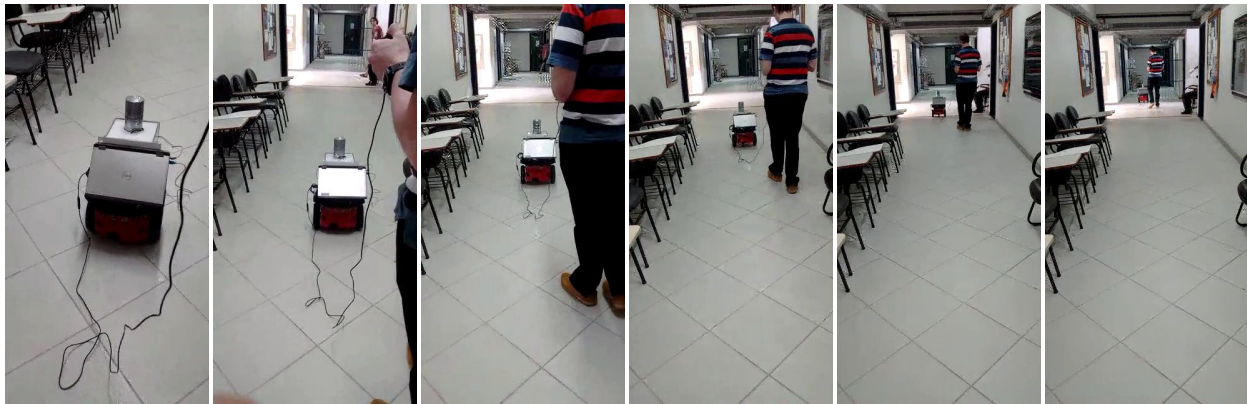
Foi criado o `process-volta_no_ct9_log_sensorbox.ini` para gravação do Log.

9 - Gravar um Log utilizando o Pioneer e a Sensorbox

Foi executado o `process-volta_no_ct9_log_sensorbox.ini` para realizar a gravação do log. Para a câmera ser gravada foi necessário alterar um parâmetro `logger save on disk` para "on" e ativar a câmera da sensorbox através do ssh.

Foi verificado que o log foi gravado com sucesso ao executar o `process-pioneer_sensorbox_playback_viewer_3D.ini`.

Para a criação do log, o controle do robô foi feito através de um operador humano usando um Joystick. O percurso inicial realizado foi do corredor do CT-9 próximo à porta do LCAD até o outro lado do corredor, conforme sequência de imagens abaixo.



10 - Função de calibração da odometria sem GPS

Foi criado um módulo `src/odometry_calibration_nogps/` para realizar a calibração da odometria sem utilizar GPS.

Como calibrar a odometria:

1. Compile o módulo `odometry_calibration`
2. Vá para o diretório bin: `cd $CARMEN_HOME/bin`
3. Execute o programa:

```
./calibrate_bias_from_log --gps_to_use 2 --use_non_linear_phi 0 -i 300
/dados/log_volta-do-ct9.txt ../src/carmen-pioneer-sensorbox.ini caco.txt poses.txt
poses_opt.txt
```

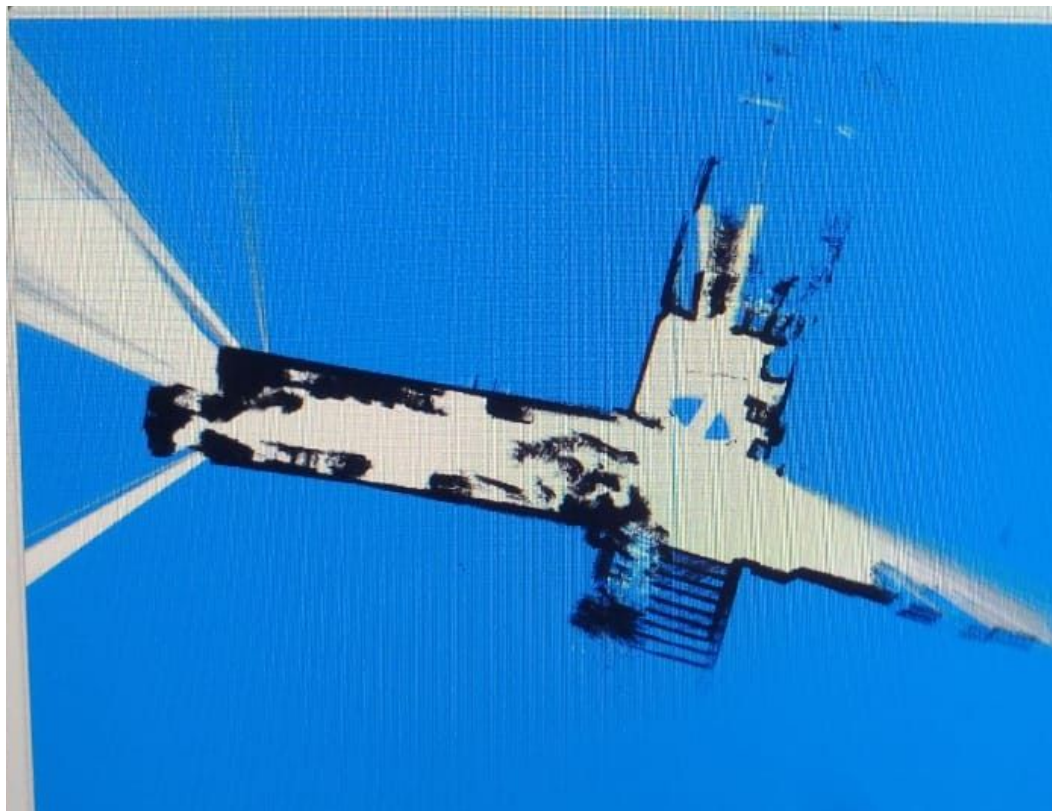
4. Pode-se usar o `poses_opt.txt` para fazer mapas sem `graphslam`! Basta usá-lo como se fosse a saída do `graphslam` para o log. Para isso, no process de fazer mapas, basta usar a linha do `graphslam_publish` como abaixo:

```
Publish_poses graphslam 1 0
./graphslam_publish ../src/odometry_calibration_nogps/poses_opt.txt
```

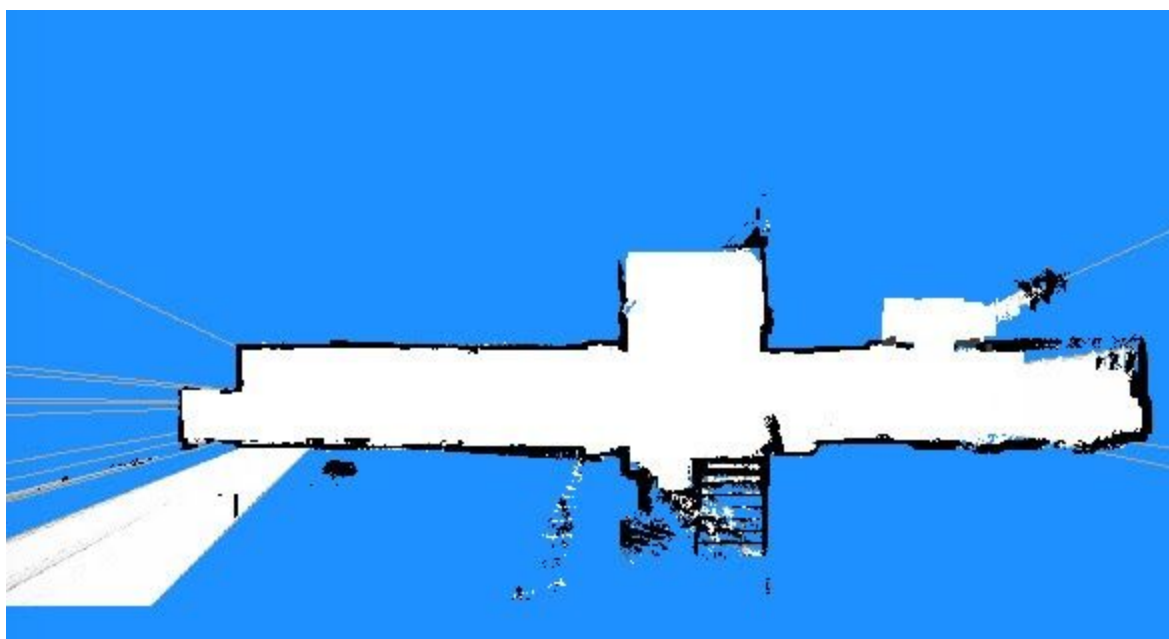
11 - Utilizar o log para criar um mapa

Utilizamos o `process-pioneer_sensorbox_playback_viewer_3D_map_generation.ini` para ajudar a criar o mapa. Depois fizemos a limpeza do mapa utilizando o `map_editor`. As poses do robô foram utilizadas para fazer o RDDL.

Mapa antes da limpeza, com todos os obstáculos:



Mapa após a limpeza utilizando a ferramenta map_editor:



12 - Ajustar parâmetros de PID

Ajustamos no carmen-pioneer-sensorbox.ini os parâmetros de PID para que os valores de integral e diferencial e correção ficassem os mais suaves possíveis para o Pioneer.

```
robot_PID_steering_kp          2000.0
robot_PID_steering_ki          100.0
robot_PID_steering_kd           0.0

robot_PID_minimum_delta_velocity      0.2

robot_PID_velocity_forward_accelerating_Kp      25.0 # 10.0
robot_PID_velocity_forward_accelerating_Ki      6.0 # 3.0 # 0.35      #10.0
robot_PID_velocity_forward_accelerating_Kd      -0.2 # -0.025 #-0.5

robot_PID_velocity_forward_deccelerating_Kp      30.0 # 20.0
robot_PID_velocity_forward_deccelerating_Ki      2.0 # 9.0 # 9.0 # 0.2
#10.0
robot_PID_velocity_forward_deccelerating_Kd      -0.2 #-0.025 #-1.0

robot_PID_velocity_backward_accelerating_Kp      10.0
robot_PID_velocity_backward_accelerating_Ki      3.0 # 0.35      #10.0
robot_PID_velocity_backward_accelerating_Kd      0.0 #-0.025 #-0.5

robot_PID_velocity_backward_deccelerating_Kp      20.0
robot_PID_velocity_backward_deccelerating_Ki      9.0 # 0.2      #10.0
robot_PID_velocity_backward_deccelerating_Kd      0.0 #-0.025 #-1.0

robot_PID_velocity_brake_gap          17.0
```

13- Fazer o pioneer se localizar e se mover autônomo em um RDDF específico

Utilizamos o `process-volta_no_ct9_log_sensorbox.ini` para fazer o pioneer se locomover autônomo. Colocamos o pioneer no início do corredor, e com o RDDF definido ele trafegou sem colidir com nenhum obstáculo. Verificamos também que, quando há um obstáculo no caminho, ele para imediatamente conforme o esperado.

Discussões e Conclusões do projeto

I. Melhorias para serem realizadas no CARMEN-LCAD para facilitar integração

Verificamos que seria bom se, no arquivo de parâmetros, ao lado de cada parâmetro estivesse comentário sobre para o que o parâmetro é utilizado e qual medida reflete o valor desse parâmetro. Verificamos que há parâmetros duplicados, mas dado que são mais de 2 milhões de linhas de código é algo normal de ocorrer. Propomos uma interface para integração de um novo robô com o carmen que seja estilo um Wizard, em que o usuário insere as medidas do robô, as medidas da sensorbox em relação ao eixo traseiro do robô, todas as medidas necessárias, e com isso o arquivo carmen-robô-sensorbox.ini seria gerado automaticamente. Com isso a integração poderia ser feita de forma mais ágil em qualquer veículo, e também pode ter um apelo comercial maior.

II. Dificuldades que tivemos

A maior dificuldade do trabalho foi sair do zero e entender o que deve ser integrado no código do pioneer. Depois que descobrimos como inserir a função de movimento com IPC para o movimento do robô, conseguimos caminhar na implementação para utilizar o robô com o joystick. Outra dificuldade que deu muito trabalho foi implementar a função de criar o mapa, visto que a função original da IARA utiliza o GPS para calibrar a odometria, e no nosso robô não há GPS disponível no ambiente indoor dado às restrições do sensor GPS.

III. Aprendizado com o mundo real

Após superadas as dificuldades, verificamos que o mundo real é um ambiente em que se deve ajustar parâmetros e verificar o comportamento do robô a todo momento. Alteramos os parâmetros de PID de forma empírica e chegamos em uma configuração adequada para o Pioneer, fazendo-o trafegar autônomo dentro do corredor do CT. Acreditamos que com esse conhecimento podemos entender e integrar outros veículos com o Carmen-LCAD.

Referências:

- 1 - Pioneer 3-DX specifications.
<https://www.generationrobots.com/media/Pioneer3DX-P3DX-RevA.pdf> . Acessado em 26/11/2019.
- 2 - Pioneer 3 - ROBOTS - YOUR GUIDE TO THE WORLD OF ROBOTICS.
<https://robots.ieee.org/robots/pioneer/> . Acessado em 26/11/2019.
- 3 - GitHub - CARMEN-LCAD. https://github.com/LCAD-UFES/carmen_lcad
Códigos-fonte disponíveis.
- 4 - CARMEN 0.7.4-beta http://carmen.sourceforge.net/getting_carmen.html
- 5 - Inter Process Communication (IPC). <http://www.cs.cmu.edu/~ipc/> . Acessado em 26/11/2019.
- 6 - HDL-32E Velodyne. <https://velodynelidar.com/hdl-32e.html> . Acessado em 27/11/2019.