groningen 2025

Positioning System for the European Hyperloop Center

Edition 2025

European Hyperloop Week

Tech Committee

EUROPEAN HYPERLOOP WEEK

This page has been left intentionally blank.

Foreword

Dear Participants,

This document describes the positioning system present in the European Hyperloop Center infrastructure. It can be used to determine both the absolute position as well as the relative position and velocity of the testing vehicles.

For inquiries, please contact the technical EHW team at technical@hyperloopweek.com.

Best of regards and good luck!

The EHW Technical Team

This page has been left intentionally blank.

Contents

All rights reserved. No part of this publication may be copied, reproduced, or distributed in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the EHW 2024-2025 Committee, except in the case of brief quotations embodied in critical reviews and certain other non-commercial uses permitted by copyright law.

January 13, 2025, the EHW Committee

This page has been left intentionally blank.

1 System Description

The positioning system for the European hyperloop center consists of two independent systems. For absolute positioning of the vehicle in the tube a barcode positioning system is used. Each barcode uniquely identifies 4mm in the infrastructure.

Below the barcode sticker a sticker suitable for an incremental encoder is placed. This system comprises of a black and white alternating bars with a pitch of 1 mm. Below an index track is created with 400 mm indices that line up with the propulsion track blocks and thus effectively with the 0 degree commutation angle. A full electrical rotation of the propulsion system coincides with 400 mm of travel along the infrastructure.

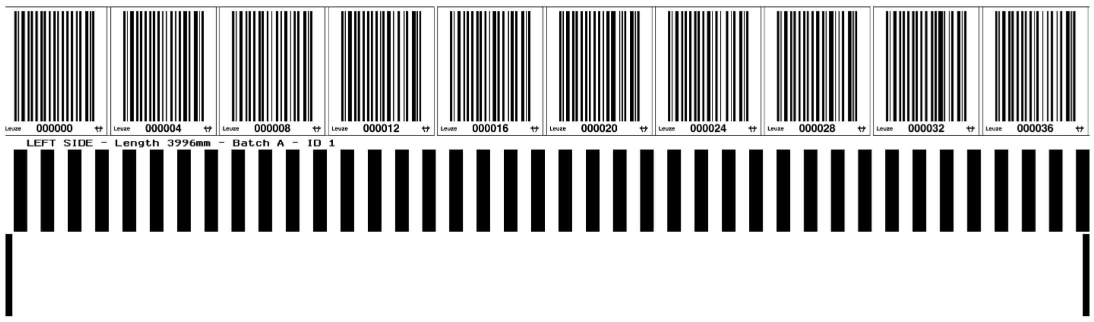

Figure 1: 400 mm Positioning Track Example.

To read the barcode sticker any barcode scanning equipment can be used. The BPS system from Leuze has been validated up to 10 m/s. This system scans four barcodes at a time and gives a position and velocity measurement.

Figure 2: Leuze BPS338i Barcode Sensor.

For higher speeds the incremental encoder track can be used. For this system a contrast sensor is suitable to output a pulse pattern coinciding with the black and white stripes.

When the incremental track is read by two sensors one can determine whether the vehicle is moving forward or backwards. With the addition of the third sensor also the 0 position relative to the magnetic field of the linear motor can be determined. These three signals can be interfaced to a Quadrature

Encoder Interface (QEI) found on many microcontrollers and industrial systems.

Below a cross-section of the infrastructure can be seen. This figure shows the positioning of the separate systems to correctly read out the position in the infrastructure.

This measurement track is placed on either side of the tube.

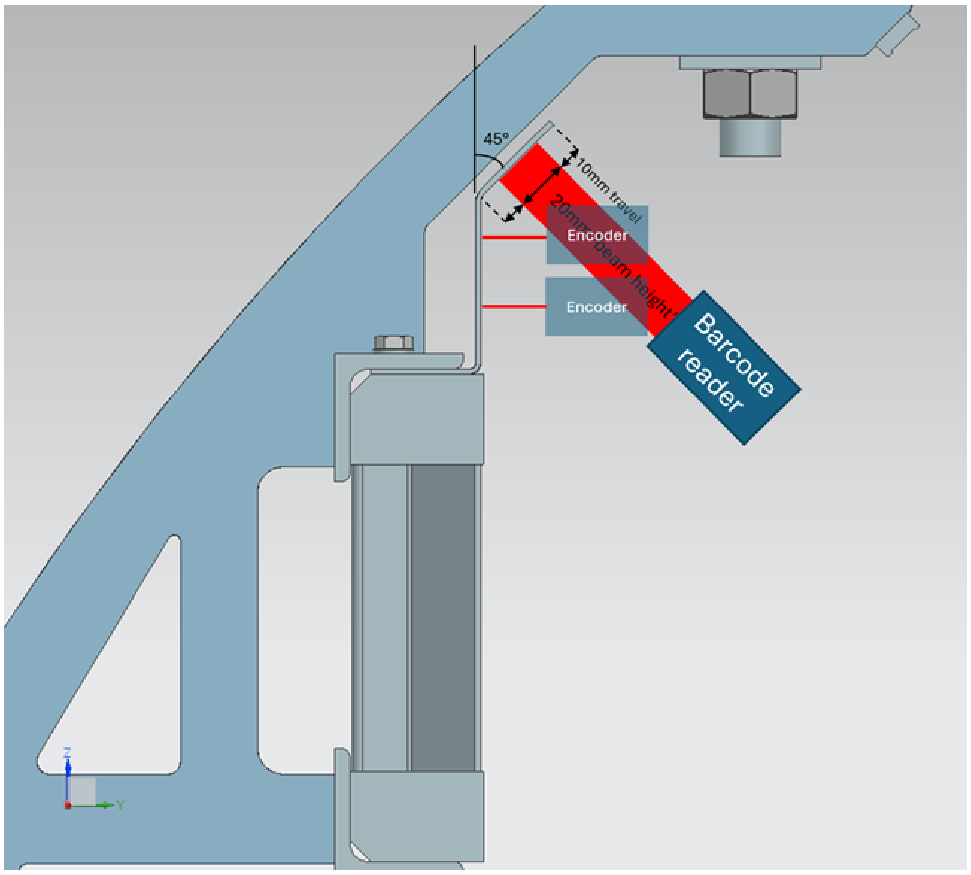

Figure 3: Positioning of sticker tracks within the infrastructure.

2 Barcode encoding

Although each barcode encodes a location within 4 mm it does not directly relate to the absolute position inside the infrastructure. This has two main reasons.

1. 1. The same stickers have been used for the left and right side of the pipe. On the left side this means that the barcodes are increasing with the direction of travel. However, on the right side this means that it is decreasing with the direction of travel.
2. 1. For logistical reasons it was impossible to install the 4 meter tracks in correct order, this means that the position that the sensor reads will jump across the valid range every 4 meter.

Additionally, the infrastructure contains expansion joints where the tracks are 4.2m long on either side to cope with the 400 mm length of the joint.

This means that the absolute position needs to be derived from the measured barcode. In Appendix A the two tables that describe the relation between barcode and absolute position can be found.

The code in Appendix B uses these tables to determine the absolute position given a barcode measurement.

This code tries to find the track in which the barcode measurement lies and to optimize the runtime performance it only looks at the previous or next track as possible other tracks to transition to. If no valid track is found it uses the vehicle velocity to estimate the position based on the previous position.

It also outputs a valid flag if the measurement was found inside a track range. This can be used if multiple sensors are available to only use sensors with valid measurements.

In case the barcode sensor travels across an expansion gap it will take the velocity of the vehicle to estimate the correct position of the vehicle. This assumes that the vehicle is not accelerating significantly between an execution cycle.

The data from the table should be formatted as the following structure (In MATLAB):

```
track.id = 1;                % Corresponds to the track number
track.start = 640;           % Starting position of the track
track.end = 644;             % Ending position of the track
track.absolute_start = 0;    % Absolute starting position in global coordinates
track.absolute_end = 4;      % Absolute ending position in global coordinates
```

3 Appendix A

track_number	x_position	type	batch	ID	Length	start_value	end_value
1	0	straight	B	61	4	640.0	644.0
2	4	straight	B	58	4	628.0	632.0
3	8	straight	B	60	4	636.0	640.0
4	12	straight	B	24	4.2	304.6	308.8
5	16.4	straight	B	2	4.2	212.2	216.4
6	20.6	straight	B	66	4	660.0	664.0
7	24.6	straight	B	65	4	656.0	660.0
8	28.6	straight	B	55	4	616.0	620.0
9	32.6	straight	B	47	4	584.0	588.0
10	36.6	straight	B	44	4	572.0	576.0
11	40.6	straight	B	178	4	1108.0	1112.0
12	44.6	straight	B	4	4.2	220.6	224.8
13	49	straight	B	9	4.2	241.6	245.8
14	53.2	straight	B	181	4	1120.0	1124.0
15	57.2	straight	B	164	4	1052.0	1056.0
16	61.2	straight	B	74	4	692.0	696.0
17	65.2	straight	B	166	4	1060.0	1064.0
18	69.2	straight	B	32	4	524.0	528.0
19	73.2	straight	B	184	4	1132.0	1136.0
20	77.2	straight	B	187	4	1144.0	1148.0
21	81.2	straight	B	16	4	460.0	464.0
22	85.2	straight	B	13	4	452.0	456.0
23	89.2	straight	B	22	4	448.0	452.0
24	93.2	straight	B	15	4	460.0	456.0
25	97.2	straight	B	158	4	1028.0	1032.0
26	101.2	straight	B	197	4	1180.0	1184.0
27	105.2	straight	B	120	4	624.0	620.0
28	109.2	straight	B	20	4.2	254.2	250.0

Table 1: Lookup Table - Starboard (Left)

track_number	x_position	type	batch	ID	Length	start_value	end_value
1	0	straight	B	52	4	608.0	604.0
2	4	straight	B	87	4	748.0	744.0
3	8	straight	B	86	4	744.0	740.0
4	12	straight	B	1	4.2	212.2	208.0
5	16.4	straight	B	16	4.2	275.2	271.0
6	20.6	straight	B	63	4	652.0	648.0
7	24.6	straight	B	64	4	648.0	644.0
8	28.6	straight	B	59	4	636.0	632.0
9	32.6	straight	B	51	4	604.0	600.0
10	36.6	straight	B	72	4	688.0	684.0
11	40.6	straight	B	69	4	676.0	672.0
12	44.6	straight	B	12	4.2	258.4	254.2
13	49	straight	B	116	4	864.0	860.0
14	53.2	straight	B	32	4	496.0	492.0
15	57.2	straight	B	33	4	492.0	488.0
16	61.2	straight	B	18	4	448.0	444.0
17	65.2	straight	B	29	4	516.0	512.0
18	69.2	straight	B	33	4	532.0	528.0
19	73.2	straight	B	186	4	1140.0	1136.0
20	77.2	straight	B	187	4	1144.0	1140.0
21	81.2	straight	B	17	4	468.0	464.0
22	85.2	straight	B	21	4	484.0	480.0
23	89.2	straight	B	23	4	480.0	476.0
24	93.2	straight	B	15	4	460.0	456.0
25	97.2	straight	B	197	4	1188.0	1184.0
26	101.2	straight	B	4	4	552.0	548.0
27	105.2	straight	B	4	4	620.0	616.0
28	109.2	straight	B	4	4	254.2	250.0

Table 2: Lookup Table - Port (Right)

4 Appendix B

```
function [track_id, barcode_position, valid_position] = barcode_estimator(barcode, track_id_prev,
barcode_prev, velocity, vehicle, tracks, reset)

track_id = track_id_prev;
barcode_position = 0;

if reset
    track_id_prev = -1;
end

% First find track_id
if (track_id_prev == -1)
    index = 1;
    track = tracks(index);
    in_range = check_range(barcode, track);

    while(~in_range)
        index = index + 1;
        if (index > length(tracks))
            barcode_position = -1;
            track_id = -1;
            break;
        end
        track = tracks(index);
        in_range = check_range(barcode, track);
    end

    if (in_range)
        track_id = track.id;
        barcode_position = calculate_position(barcode, track);
    end
else
    [track_id, barcode_position] = find_position(barcode, track_id, barcode_prev, velocity, vehicle,
tracks);
end

if(track_id == -1)
    valid_position = false;
else
    valid_position = check_valid(barcode, tracks(track_id));
end

end
```

```
function [track_id, barcode_pos] = find_position(barcode, track_id, barcode_prev, velocity, vehicle,
tracks)
    % Check neighbours
    if(track_id == 1)
        track_id_prev = track_id;
        track_id_next = track_id + 1;
        track_id_current = track_id;
    elseif(track_id == length(tracks))
        track_id_prev = track_id - 1;
        track_id_next = track_id;
        track_id_current = track_id;
    else
        track_id_prev = track_id - 1;
        track_id_next = track_id + 1;
        track_id_current = track_id;
    end

    track_prev = tracks(track_id_prev);
    track_next = tracks(track_id_next);
    track_current = tracks(track_id_current);

    distance_prev = abs(barcode_prev - track_prev.absolute_end);
    distance_next = abs(barcode_prev - track_next.absolute_start);

    if (track_prev.id == track_current.id)
        distance_prev = 5000;
    end

    if (track_next.id == track_current.id)
        distance_next = 5000;
    end

    % previous track closer
    if distance_prev <= distance_next
        track_other = track_prev;
    % next track closer
    else
       track_other = track_next;
    end

    if check_range(barcode, track_other)
        track_id = track_other.id;
        barcode_pos = calculate_position(barcode, track_other);
    % in range of current
    elseif check_range(barcode, track_current)
        track_id = track_current.id;
        barcode_pos = calculate_position(barcode, track_current);
    % lost sync assume current
    else
        track_id = track_current.id;
        barcode_pos = barcode_prev + velocity * (1 / vehicle.software.operating_frequency);
    end
end
```

```
function valid = check_valid(barcode, track)
    % Starboard
    if(track.start < track.end)
        valid = barcode >= track.start + 0.2 && barcode <= track.end - 0.2;
    % Port
    else
        valid = barcode <= track.start - 0.2 && barcode >= track.end + 0.2;
    end
end

function is_in_range = check_range(barcode, track)
    % Starboard
    if(track.start < track.end)
        is_in_range = barcode >= track.start + 0.05 && barcode <= track.end - 0.05;
    % Port
    else
        is_in_range = barcode <= track.start - 0.05 && barcode >= track.end + 0.05;
    end
end

function position = calculate_position(barcode, track)
    % Starboard
    if(track.start < track.end)
        position = track.absolute_start + (barcode - track.start);
    % Port
    else
        position = track.absolute_start + (track.start - barcode);
    end
end
```