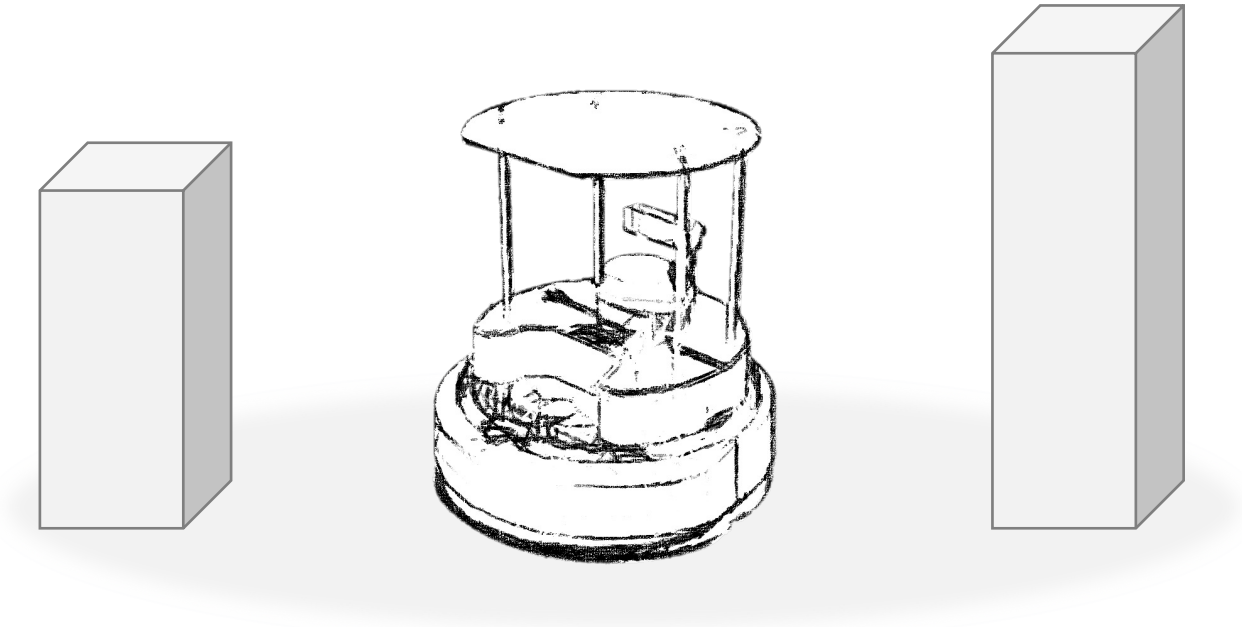
Is it a Bug?
Understanding Physical Unit Mismatches in Robotic Systems

Paulo Canelas

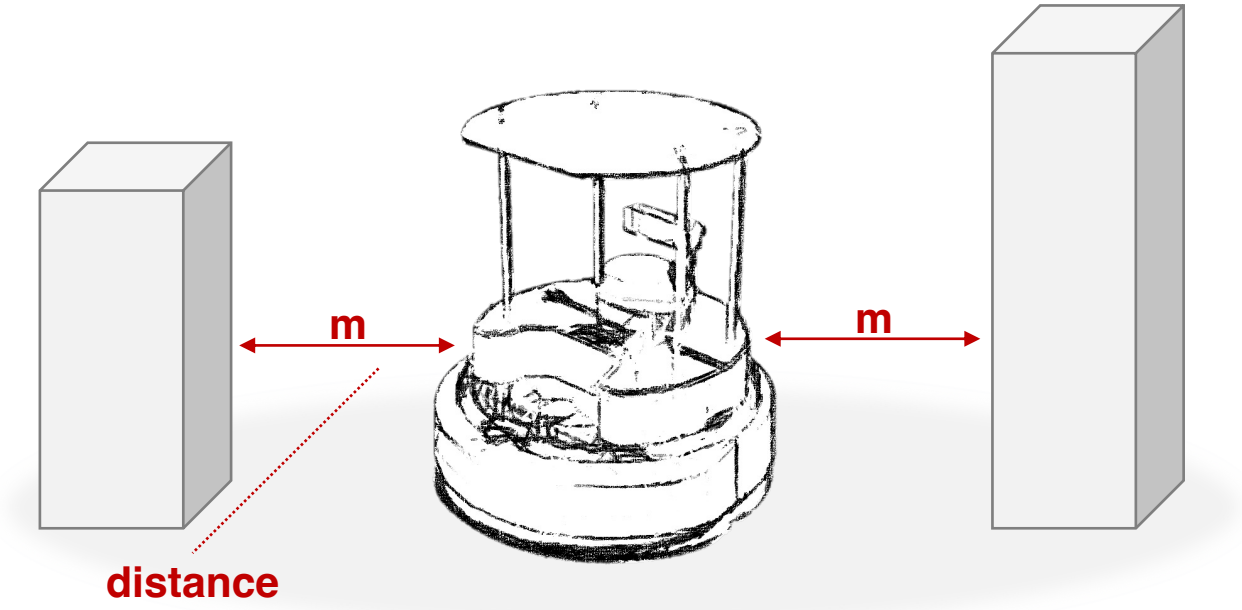
with Trenton Tabor, John-Paul Ore, Alcides Fonseca, Claire Le Goues, and Christopher S. Timperley

International Conference in Robotics and Automation (ICRA) 2024.

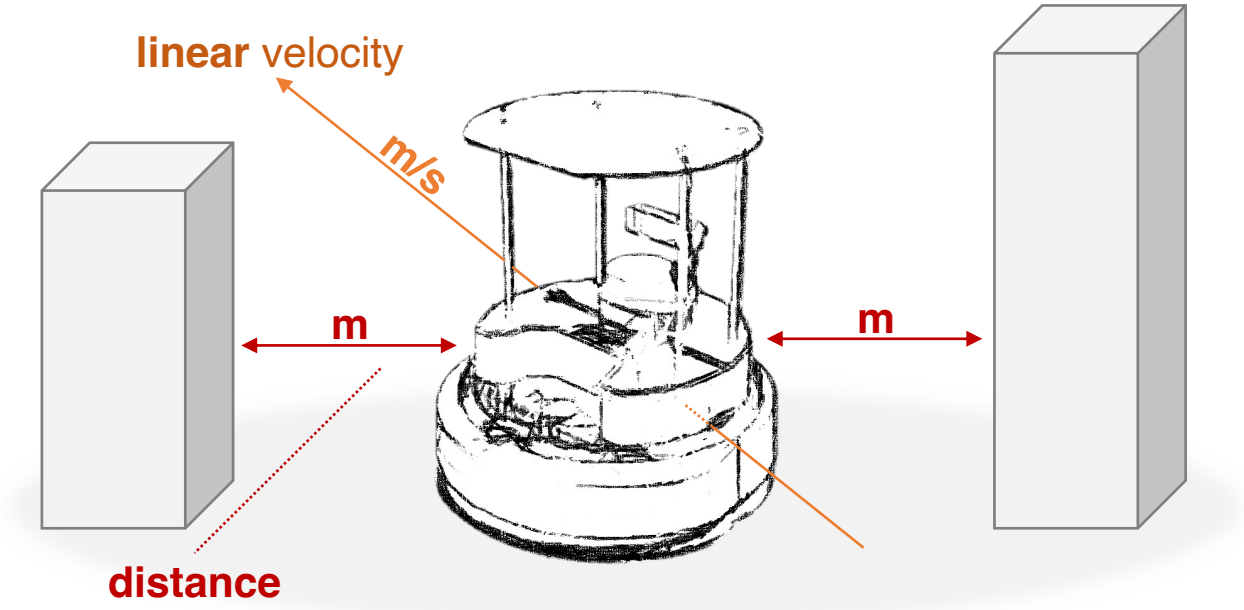
Robot code abounds with variables that are quantified using physical units



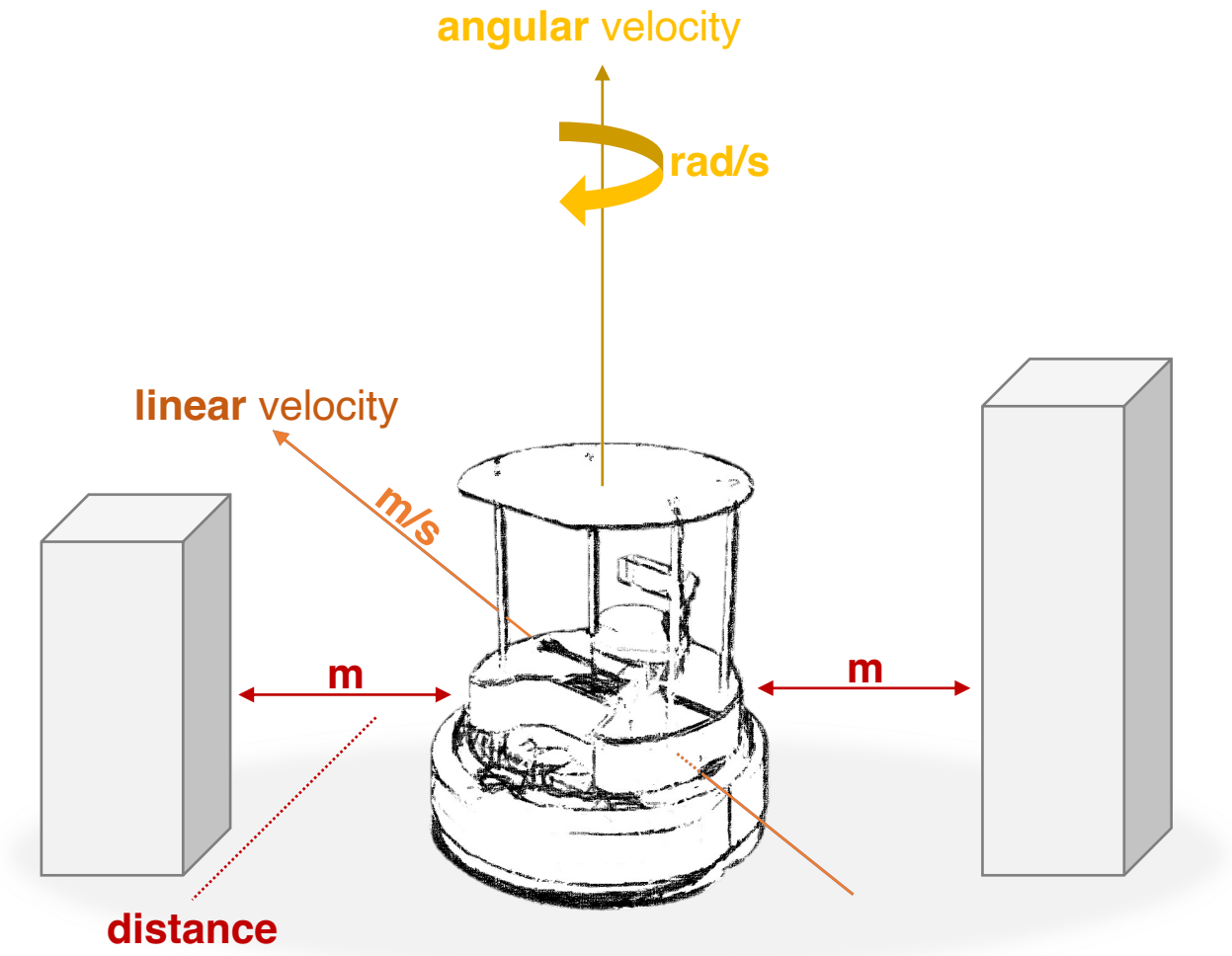
Robot code abounds with variables that are quantified using physical units



Robot code abounds with variables that are quantified using physical units



Robot code abounds with variables that are quantified using physical units

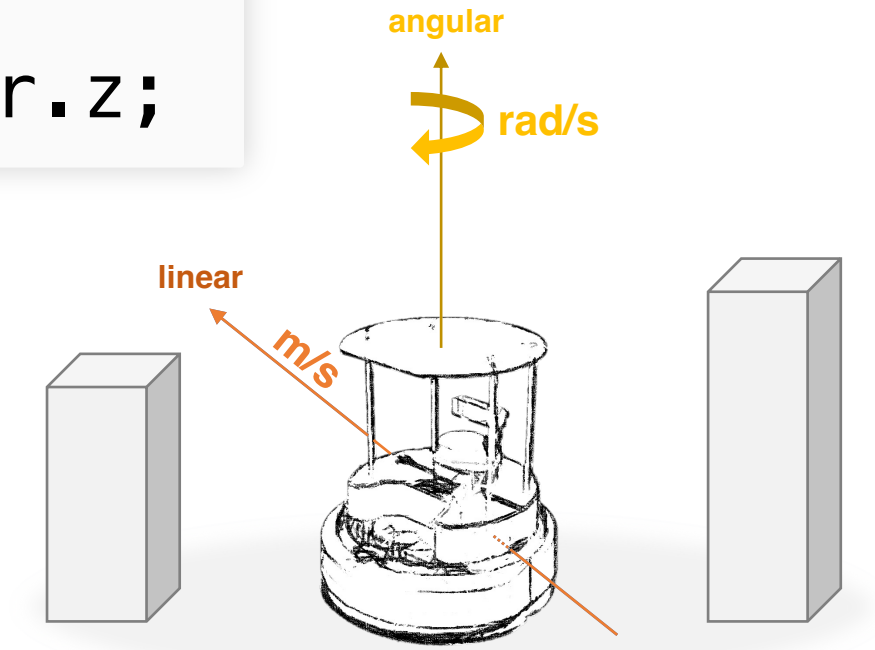


Physical unit mismatches occurs when performing incorrect operations according to dimensional analysis

```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```

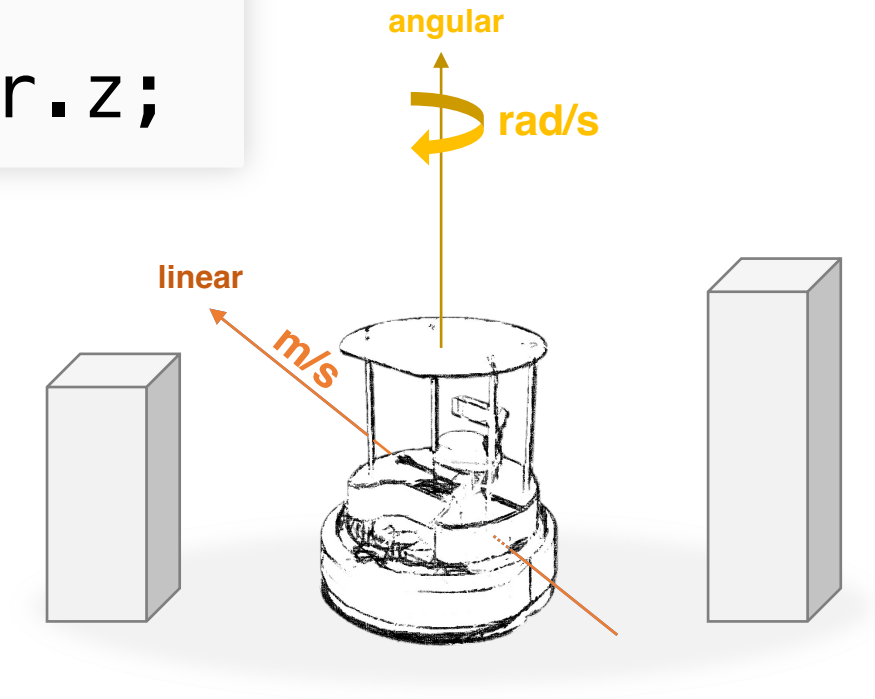
Physical unit mismatches occurs when performing incorrect operations according to dimensional analysis

```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```



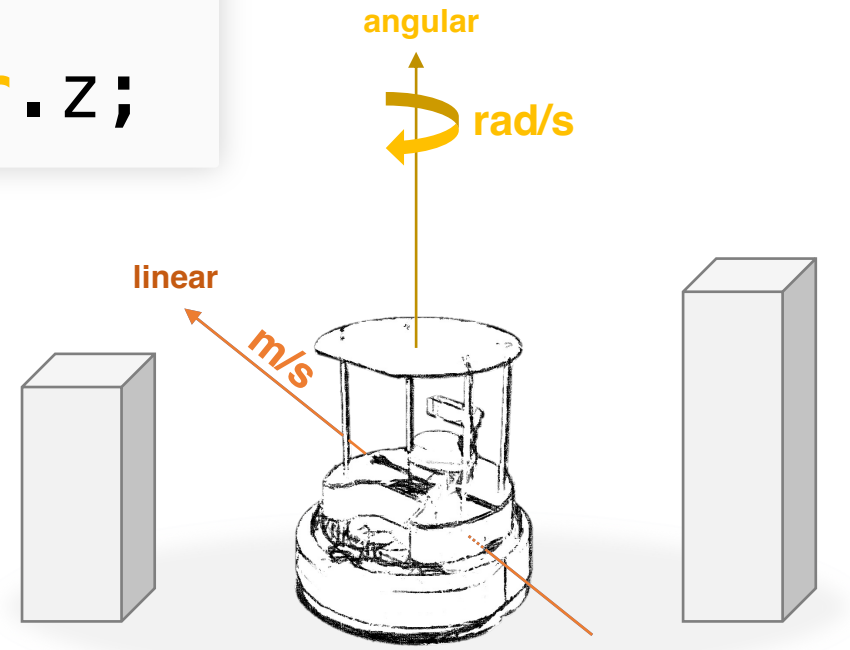
Physical unit mismatches occurs when performing incorrect operations according to dimensional analysis

```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```



Physical unit mismatches occurs when performing incorrect operations according to dimensional analysis

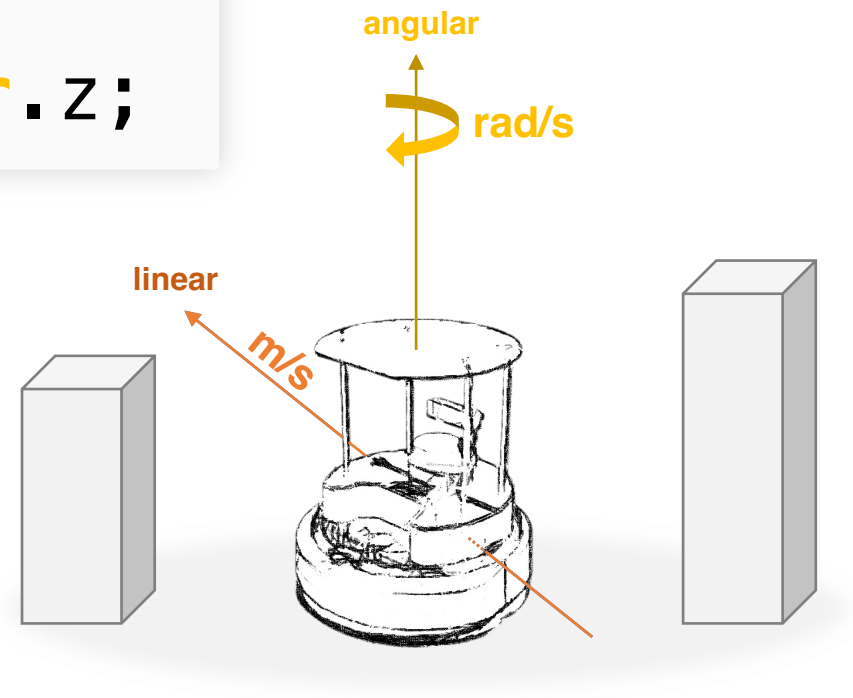
```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```



Physical unit mismatches occurs when performing incorrect operations according to dimensional analysis

```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```

Physical unit mismatch



What types of unit mismatches developers make? Long story short…

What types of unit mismatches developers make? Long story short…

We found that in the robotics domain, developers perform **both** **unintentional** and **intentional** unit mismatches

Methodology

The large codebase makes it challenging to manually search for unit mismatches

turtlebot_example_node.cpp

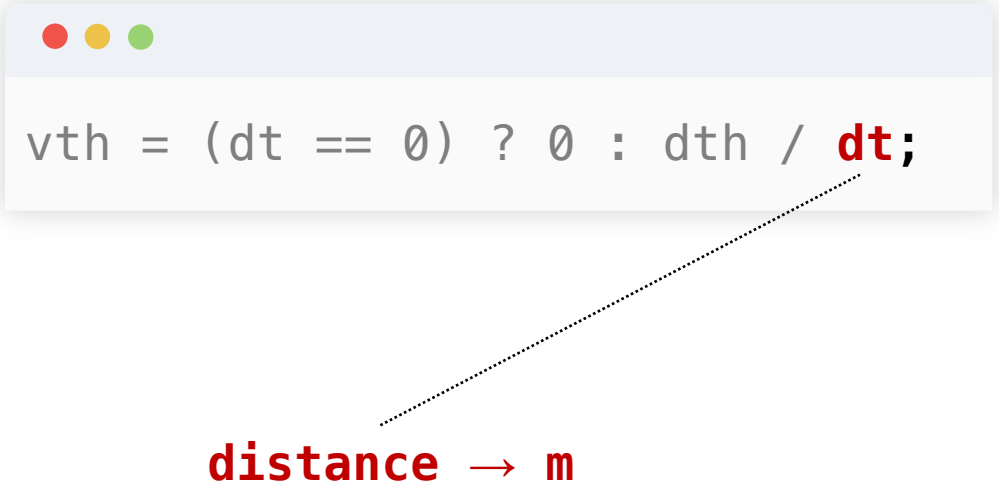
data > FrenchVanilla > src > turtlebot_example > src > turtlebot_example_node.cpp

```
28      err_x=x_t-X;
29      err_y=y_t-Y;
30      err_d=sqrt(err_x*err_x+err_y+err_y);
31      err_yaw=t-yaw_degrees;
32      //if(err_yaw<0)err_yaw=err_yaw+360;
33
34      if(err_x>0.1||err_y>0.1)
35      {vel_x=0.3;ang_z=0.0;}//ROS_INFO("1..S=%d",state);}
36      else if(err_yaw>2)
37      {ang_z=0.15;vel_x=0.0;}//ROS_INFO("2..S=%d",state);}
38      else
39      {
40          state=(state+1)%8;
41          ang_z=0.0;vel_x=0.0;
42          ROS_INFO("3..S=%d",state);
43      }
44
45  }
46
47  //Callback function for the Position topic
48  void pose_callback(const geometry_msgs::PoseWithCovarianceStamped& msg)
49  {
50      //This function is called when a new pose message is received
51
52      X = msg.pose.pose.position.x; // Robot X psotition
53      Y = msg.pose.pose.position.y; // Robot Y psotition
54      double Yaw = tf::getYaw(msg.pose.pose.orientation); // Robot Yaw
```

Phys uses names and ROS assumptions to infer the physical units and detect unit mismatches

```
vth = (dt == 0) ? 0 : dth / dt;
```

Phys uses names and ROS assumptions to infer the physical units and detect unit mismatches



Phys uses names and ROS assumptions to infer the physical units and detect unit mismatches

```
vth = (dt == 0) ? 0 : dth / dt;
```

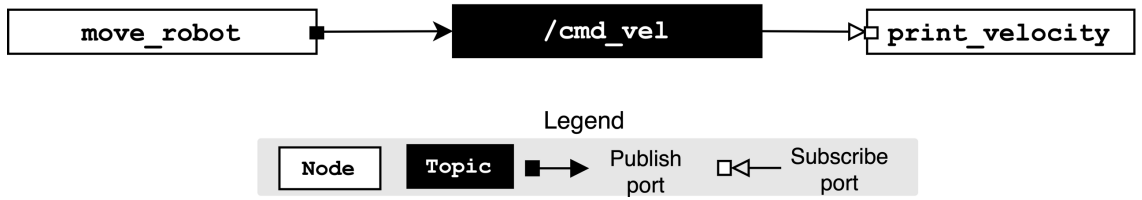
distance → m

Phys uses names and ROS assumptions to infer the physical units and detect unit mismatches

```
vth = (dt == 0) ? 0 : dth / dt;
```

distance → m

The Robot Operating System (ROS) allows developers to use publisher-subscribers to exchange messages through topics

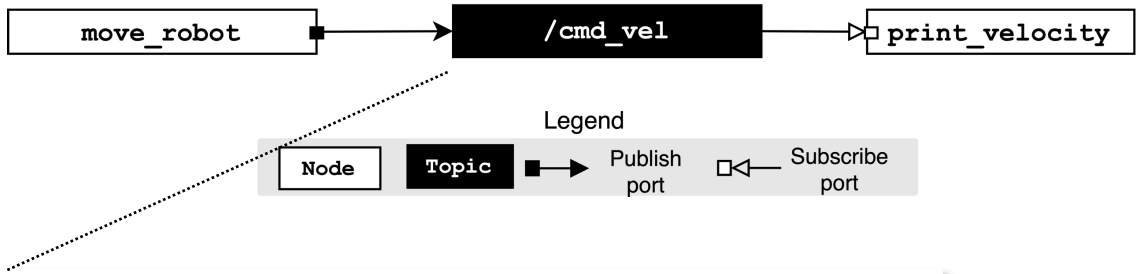


Phys uses names and ROS assumptions to infer the physical units and detect unit mismatches

```
vth = (dt == 0) ? 0 : dth / dt;
```

distance → m

The Robot Operating System (ROS) allows developers to use publisher-subscribers to exchange messages through topics



geometry_msgs/Twist Message

File: geometry_msgs/Twist.msg

Raw Message Definition

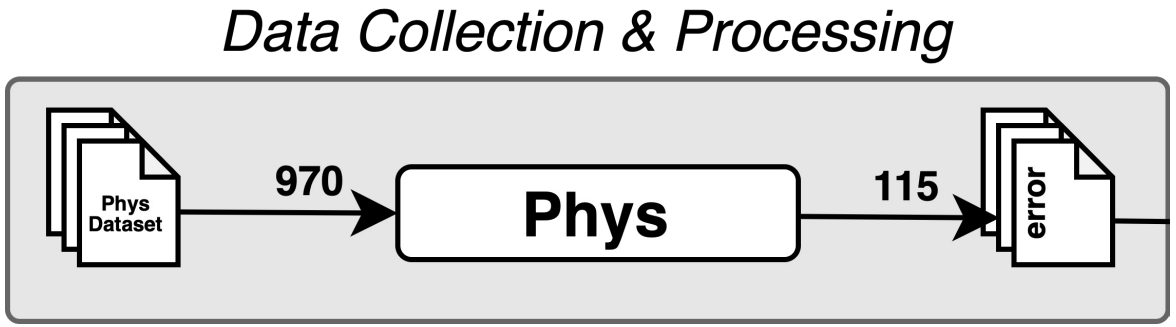
```
# This expresses velocity in free space broken into its linear and angular parts.
Vector3  linear
Vector3  angular
```

Compact Message Definition

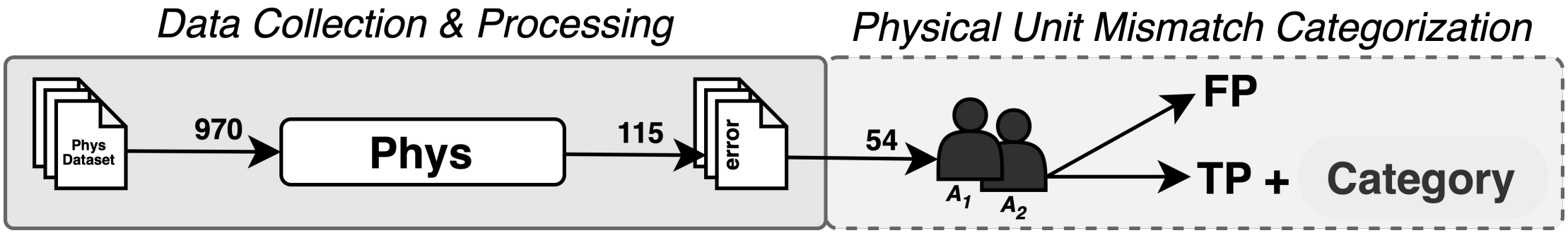
```
geometry_msgs/Vector3 linear
geometry_msgs/Vector3 angular
```

autogenerated on Wed, 02 Mar 2022 00:06:53

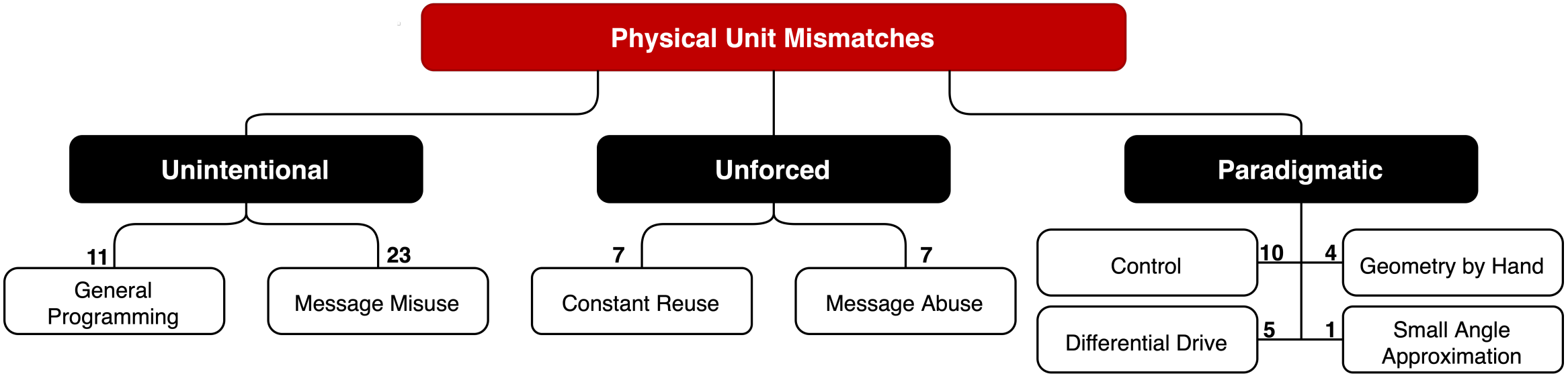
We executed Phys on its dataset and obtained 115 files with errors



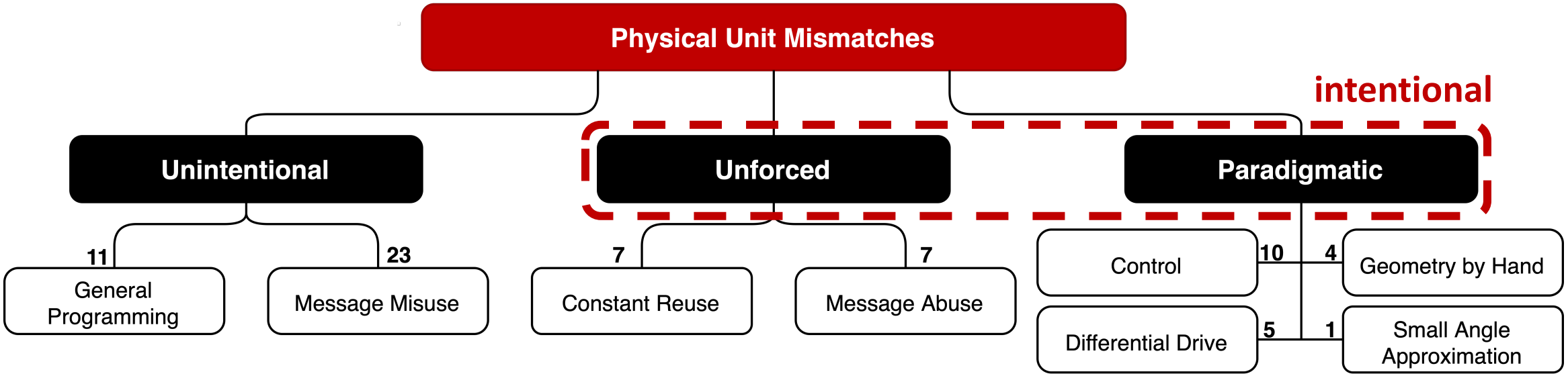
We manually analyzed 180 errors, validated with two authors and created a taxonomy



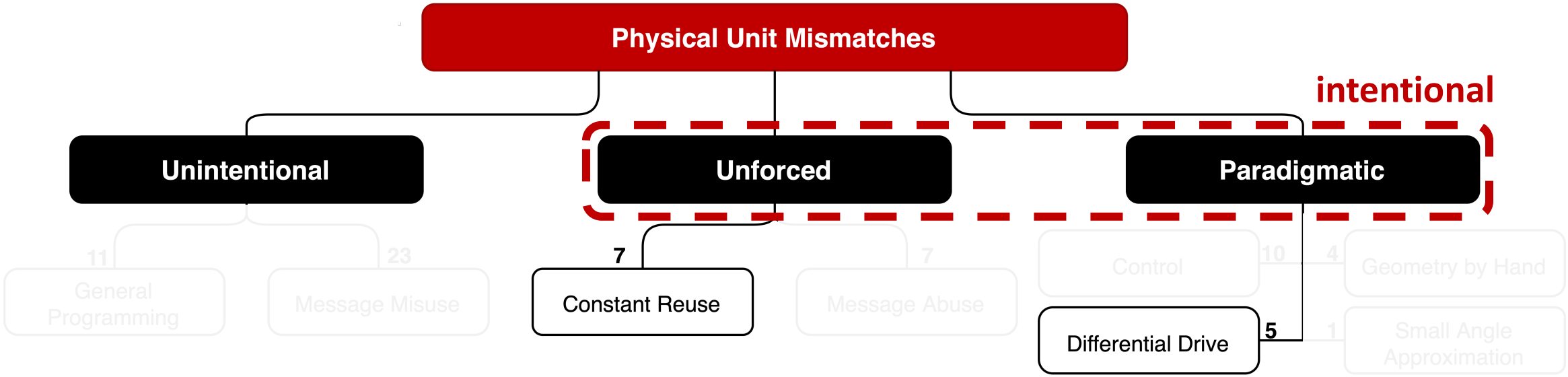
We encountered three high-level categories and eight sub-categories of unit mismatches



We encountered three high-level categories and eight sub-categories of unit mismatches



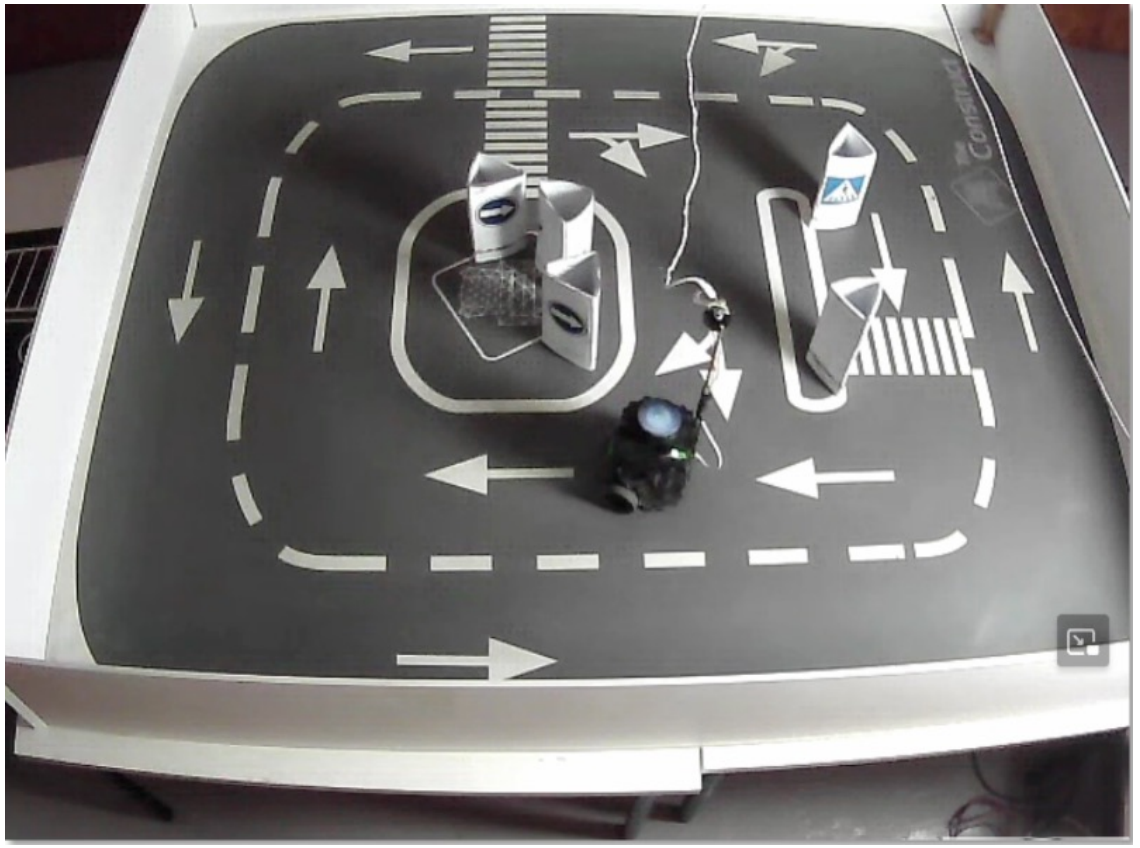
We encountered three high-level categories and eight sub-categories of unit mismatches



Unforced Unit Mismatches

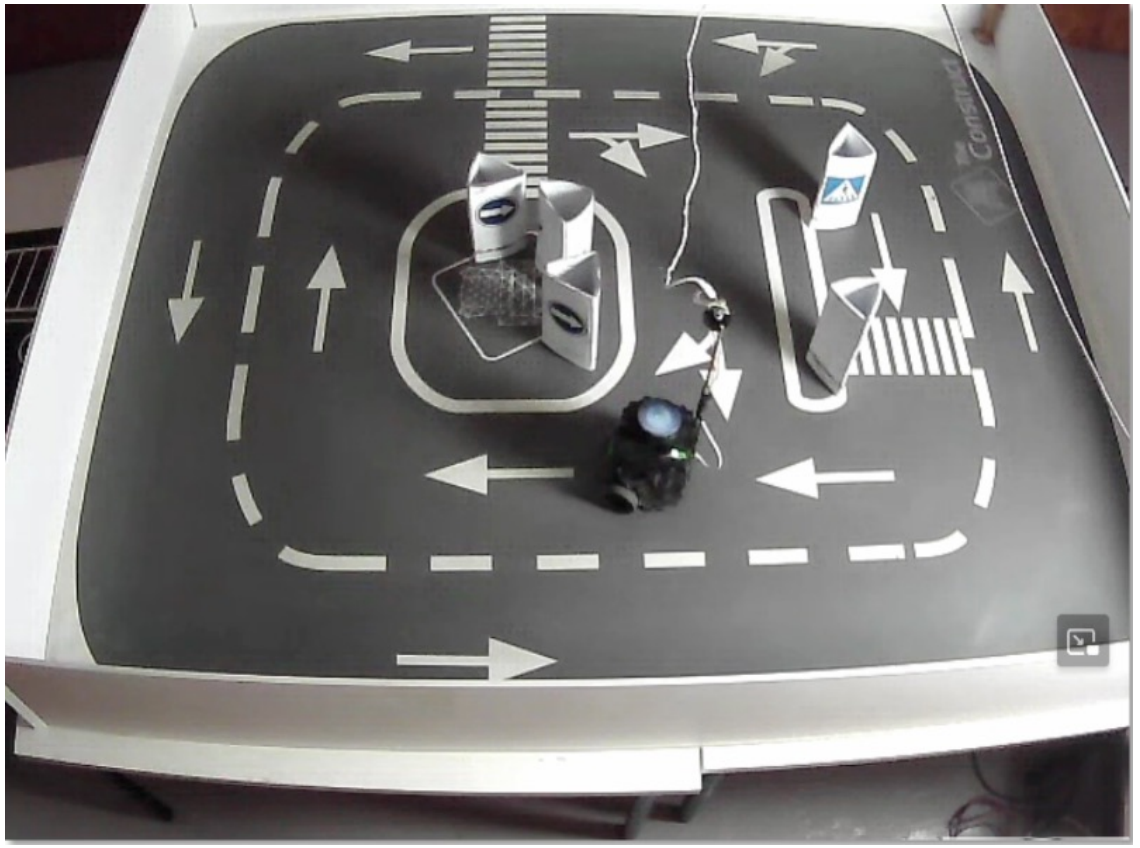
Developers reuse constants and errors to perform calculations and comparisons

```
if (cmd.linear.x > max_vel)
    cmd.linear.x = max_vel;

else if (cmd.linear.x < -max_vel)
    cmd.linear.x = -max_vel;



if (cmd.angular.z > max_vel)
    cmd.angular.z = max_vel;

else if (cmd.angular.z < -max_vel)
    cmd.angular.z = -max_vel;
```



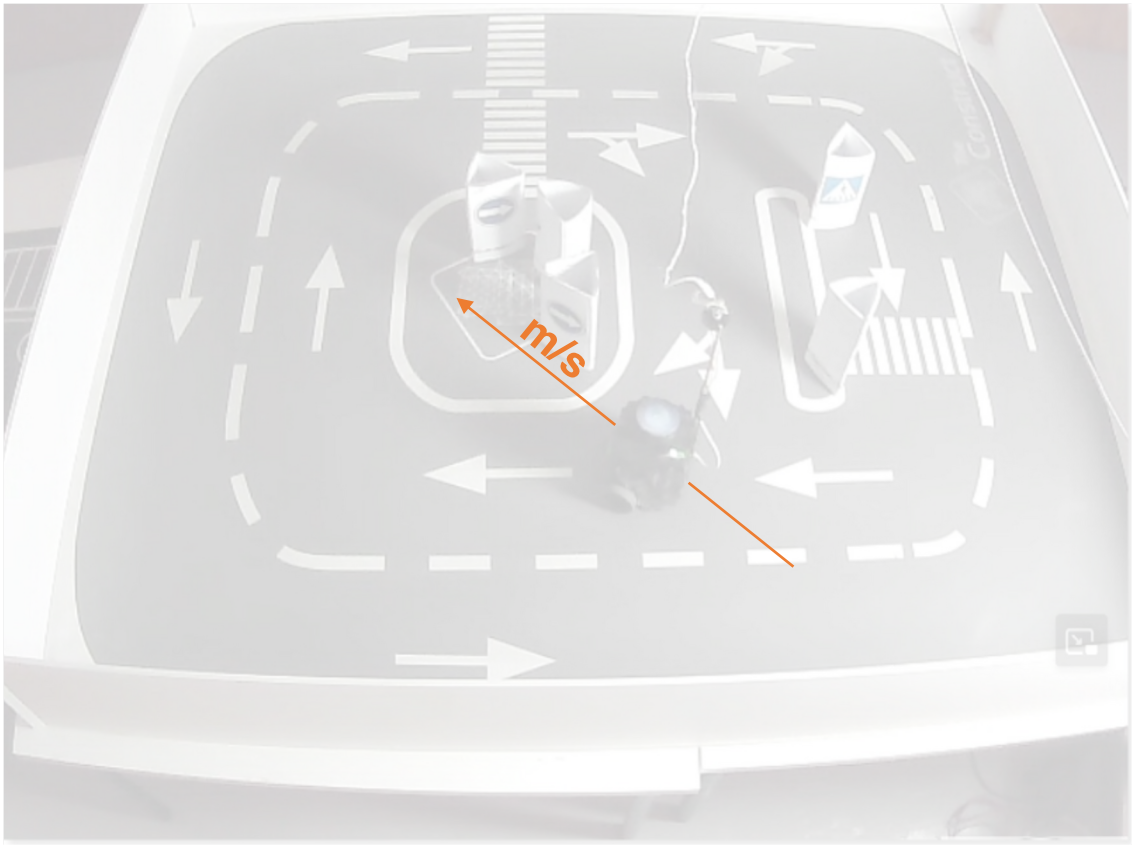
Developers reuse constants and errors to perform calculations and comparisons

```
if (cmd.linear.x > max_vel)
    cmd.linear.x = max_vel;

else if (cmd.linear.x < -max_vel)
    cmd.linear.x = -max_vel;



if (cmd.angular.z > max_vel)
    cmd.angular.z = max_vel;

else if (cmd.angular.z < -max_vel)
    cmd.angular.z = -max_vel;
```



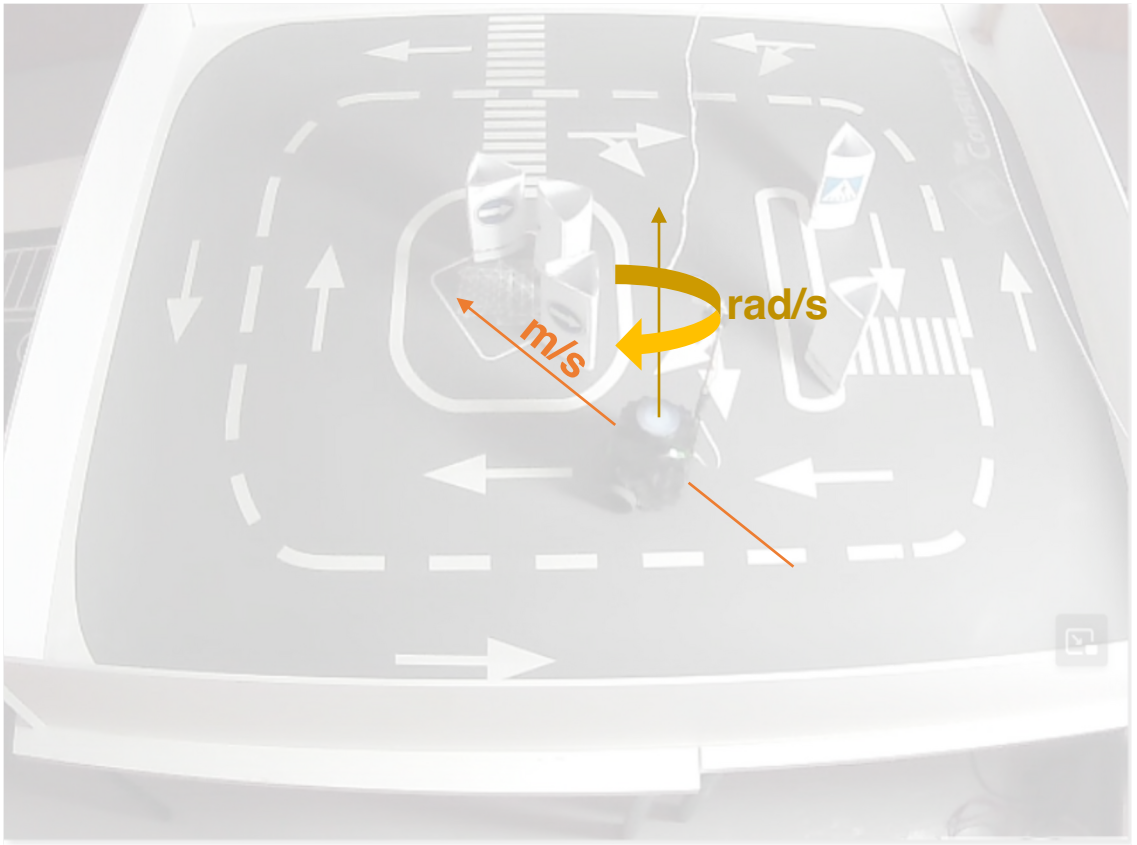
Developers reuse constants and errors to perform calculations and comparisons

```
if (cmd.linear.x > max_vel)
    cmd.linear.x = max_vel;

else if (cmd.linear.x < -max_vel)
    cmd.linear.x = -max_vel;



if (cmd.angular.z > max_vel)
    cmd.angular.z = max_vel;

else if (cmd.angular.z < -max_vel)
    cmd.angular.z = -max_vel;
```



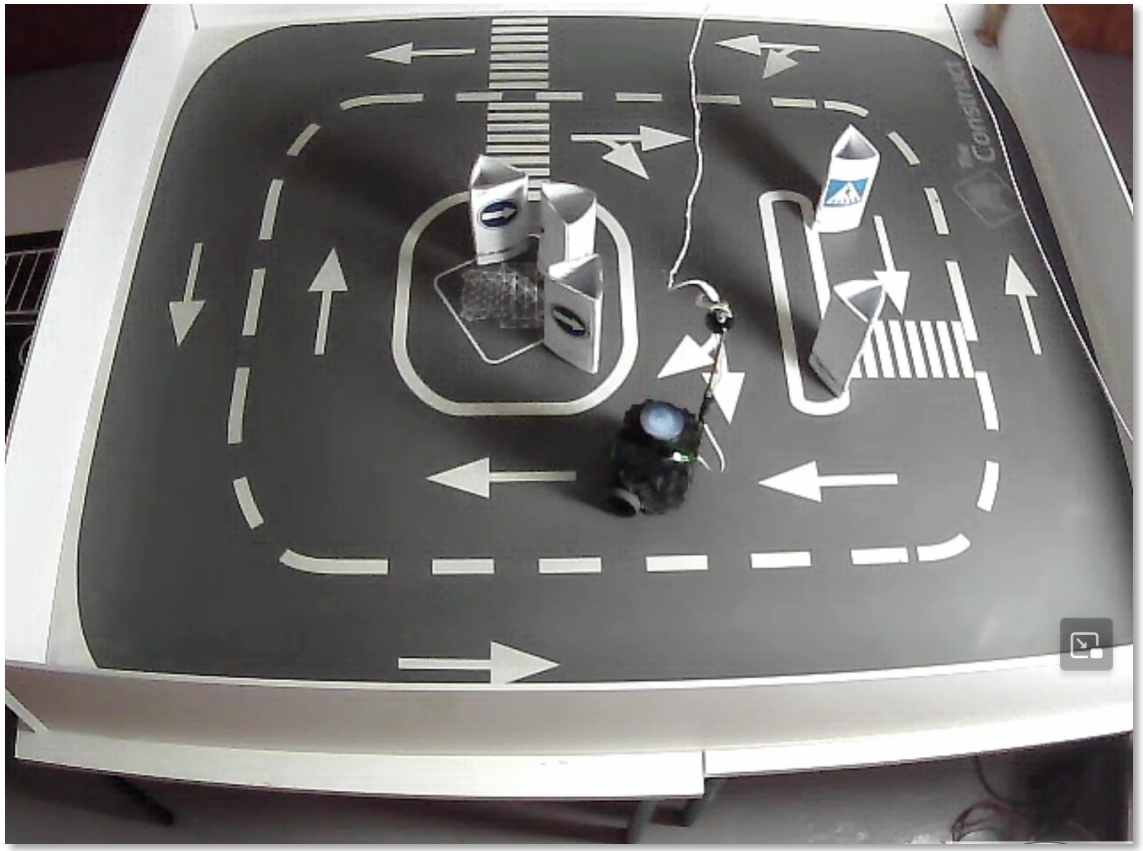
Developers reuse constants and errors to perform calculations and comparisons

```
if (cmd.linear.x > max_vel)
    cmd.linear.x = max_vel;

else if (cmd.linear.x < -max_vel)
    cmd.linear.x = -max_vel;



if (cmd.angular.z > max_vel)
    cmd.angular.z = max_vel;

else if (cmd.angular.z < -max_vel)
    cmd.angular.z = -max_vel;
```



Developers reuse constants and errors to perform calculations and comparisons

```
if (cmd.linear.x > max_vel)
    cmd.linear.x = max_vel;

else if (cmd.linear.x < -max_vel)
    cmd.linear.x = -max_vel;



if (cmd.angular.z > max_vel)
    cmd.angular.z = max_vel;

else if (cmd.angular.z < -max_vel)
    cmd.angular.z = -max_vel;
```

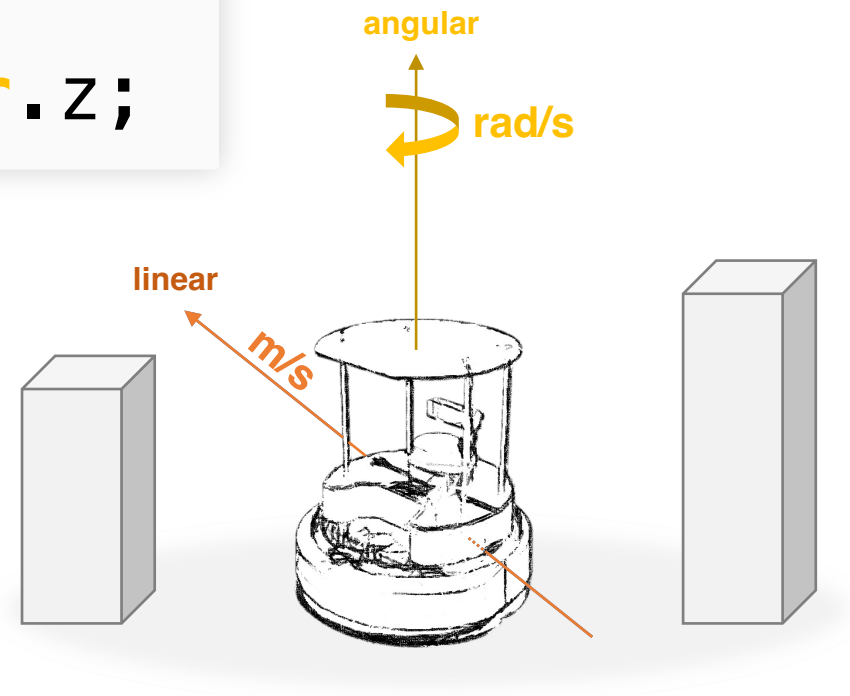


Paradigmatic Unit Mismatches

Differential Drive manifests when working with differential wheeled robots

Differential Drive manifests when working with differential wheeled robots

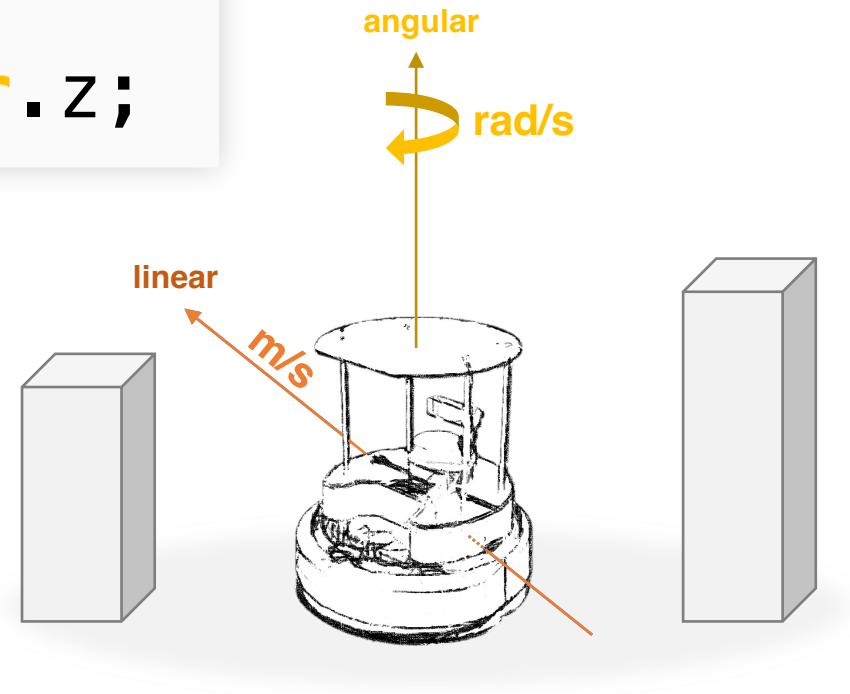
```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```



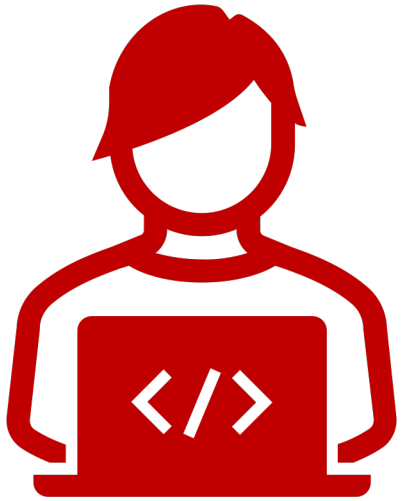
Differential Drive manifests when working with differential wheeled robots

```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```

Physical unit mismatch

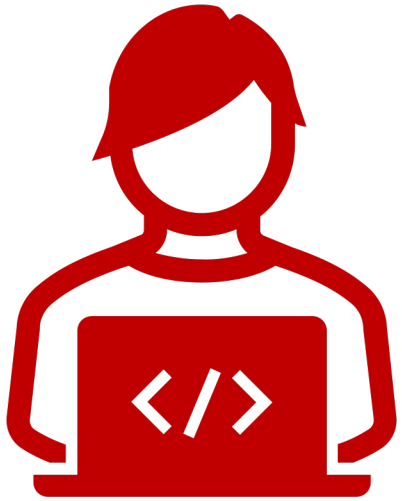


By understanding how robot developers introduce physical unit mismatches we can…



Inform the teaching of robots software development to avoid unforced unit mismatches

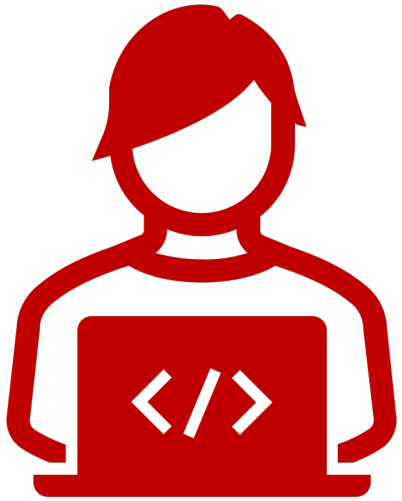
By understanding how robot developers introduce physical unit mismatches we can…



Inform the teaching of robots software development to avoid unforced unit mismatches

Improve tooling that **effectively** distinguishes the different types of unit mismatches

By understanding how robot developers introduce physical unit mismatches we can…

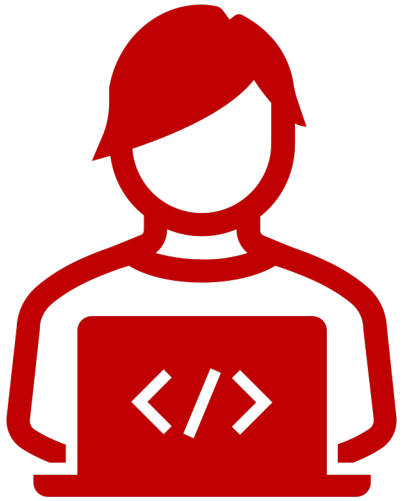


Inform the teaching of robots software development to avoid unforced unit mismatches

Multiple File Analysis

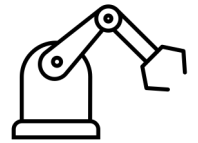
Improve tooling that **<u>effectively</u>** distinguishes the different types of unit mismatches

By understanding how robot developers introduce physical unit mismatches we can…



Inform the teaching of robots software development to avoid unforced unit mismatches

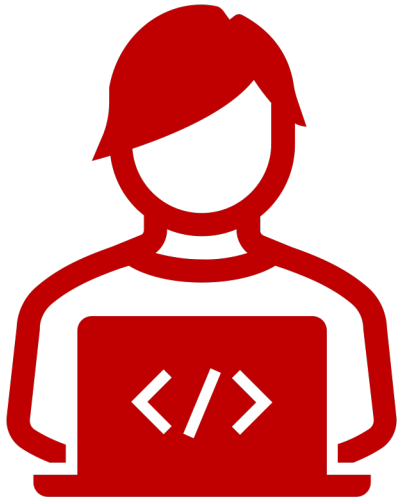
Multiple File Analysis



Context-dependent Analysis

Improve tooling that **effectively** distinguishes the different types of unit mismatches

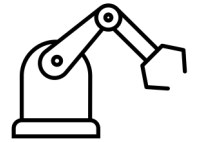
By understanding how robot developers introduce physical unit mismatches we can…



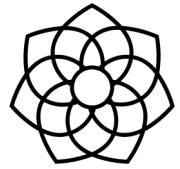
Inform the teaching of robots software development to avoid unforced unit mismatches

Improve tooling that **effectively** distinguishes the different types of unit mismatches

Multiple File Analysis



Context-dependent Analysis



Domain-specific patterns

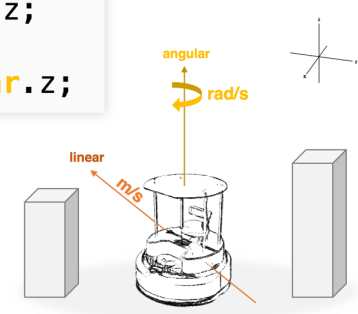
Is it a Bug?
Understanding Physical Unit Mismatches in Robotic Systems

Paulo Canelas
with Trenton Tabor, John-Paul Ore, Alcides Fonseca, Claire Le Goues, and Christopher S. Timperley
Accepted at the International Conference in Robotics and Automation 2024.

Physical unit mismatches occurs when performing incorrect operations according to dimensional analysis

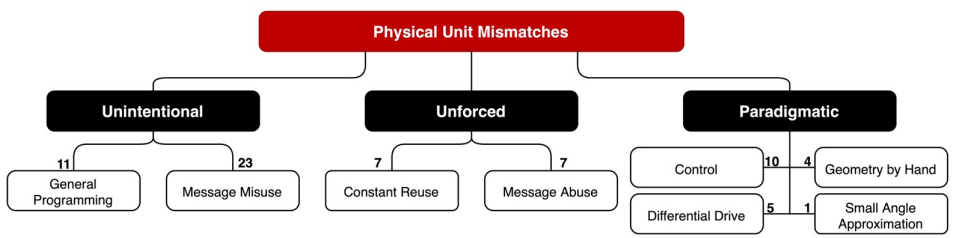
```
left_wheel = linear.x - angular.z;

right_wheel = -linear.x - angular.z;
```

Physical unit mismatch



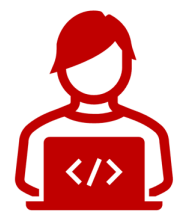
10

We encountered three high-level categories and eight sub-categories of unit mismatches



21

By understanding how robot developers introduce physical unit mismatches we can…



Inform the teaching of robots software development to avoid unforced unit mismatches

Improve tooling that **effectively** distinguishes the different types of unit mismatches

28

This work was supported by Fundação para a Ciência e Tecnologia (FCT) in the LASIGE Research Unit under the ref. (UIDB/00408/2020, UIDP/00408/2020 and EXPL/CCI-COM/1306/2021), the CMU-Portugal Dual PhD program (SFRH/BD/151469/2021), and NSF-USDA-NIFA #2021-67021-33451. The authors would like to thank Bogdan Vasilescu for his feedback on this work.