

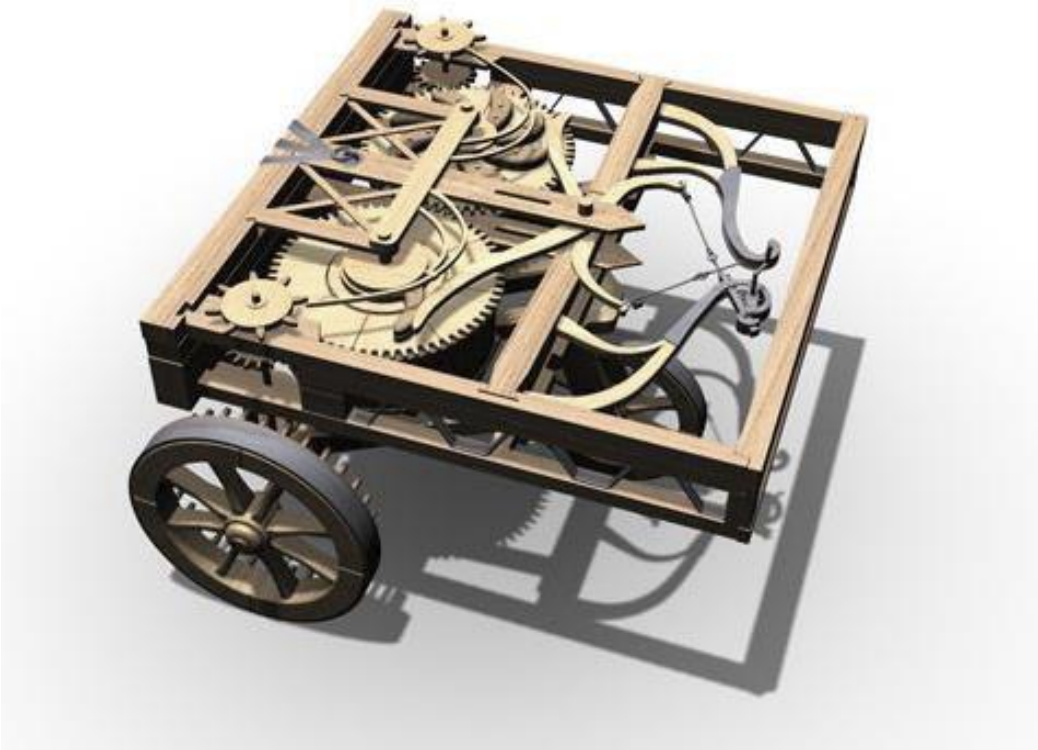
COURSE General Review 1

MOBILE ROBOTICS

CENG 5437, CENG 4391 SPRING 2023

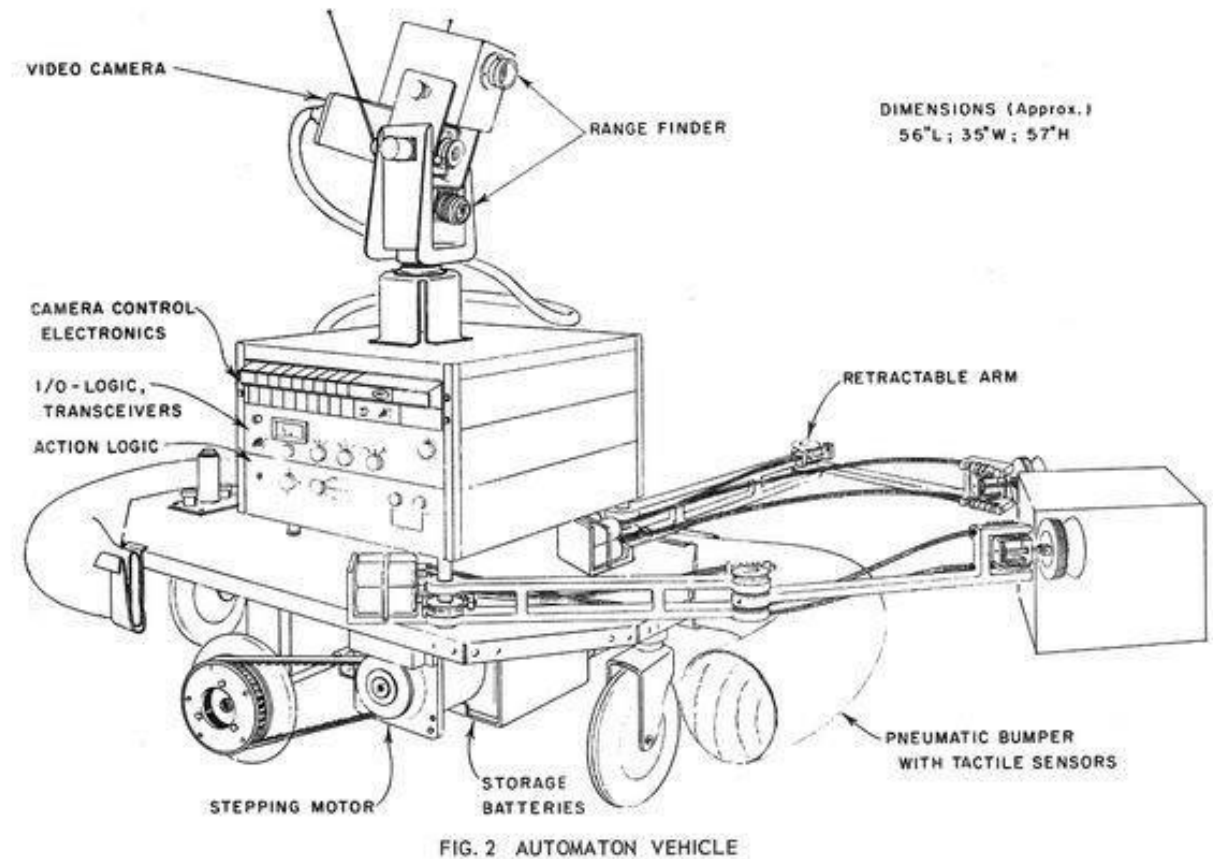
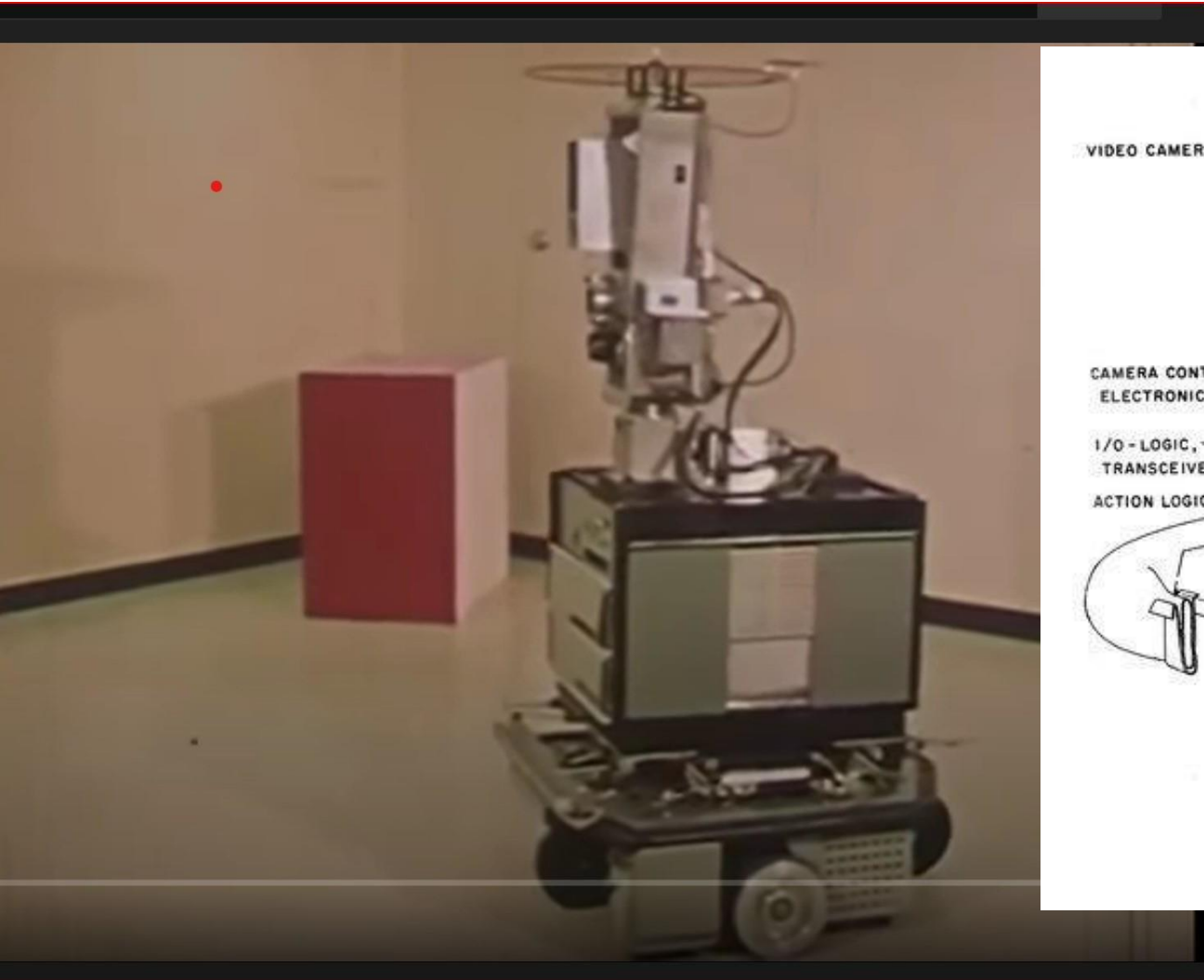


1. Introduction to the Course To the videos – History, Cars, Applications.
ROS robots
2. Physics, Inertia, URDF models
3. Selecting Motors for Wheeled Robots
4. PID control of Wheeled Robots



<http://www.universalleonardo.org/worklarge.php?id=512&image=0&trail=0&trailCount=&name=>

SHAKEY STARTS THE REVOLUTION!



The Intelligent one!!

1. Automated Delivery

<https://uh.edu/news-events/stories/2019/november-2019/11102019-starship-robots.php>

Starship Autonomous Food Delivery Robots Deployed at University of Houston



May be changed 2022 - Now Robot delivery in one of the restaurants at UH.

Good Info



StarshipPatentUS10732641.pdf

<https://patentimages.storage.googleapis.com/39/5e/9e/553ed587b6990>

1

</US10732641.pdf>

(12) **United States Patent**
Heinla et al.

(10) **Patent No.:** **US 10,732,641 B2**

(45) **Date of Patent:** **Aug. 4, 2020**

(54) **MOBILE ROBOT SYSTEM AND METHOD FOR GENERATING MAP DATA USING STRAIGHT LINES EXTRACTED FROM VISUAL IMAGES**

(71) Applicant: **Starship Technologies OÜ**, Tallinn (EE)

(72) Inventors: **Ahti Heinla**, Tallinn (EE);
Kalle-Rasmus Volkov, Tallinn (EE);
Lindsay Roberts, Tallinn (EE); **Indrek Mandre**, Tallinn (EE)

(73) Assignee: **STARSHIP TECHNOLOGIES OÜ**, Tallinn (EE)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **15/968,802**

(22) Filed: **May 2, 2018**

(58) **Field of Classification Search**

CPC .. G05D 1/0274; G05D 1/0212; G05D 1/0088;
G05D 1/0251; G05D 2201/0216;
(Continued)

(56) **References Cited**

U.S. PATENT DOCUMENTS

7,015,831 B2 3/2006 Karlsson et al.
7,873,448 B2 1/2011 Takeda et al.
(Continued)

FOREIGN PATENT DOCUMENTS

WO WO 99/30270 6/1999

OTHER PUBLICATIONS

Gaspar et al., "Vision-based Navigation and Environmental Representations with an Omni-directional Camera," IEEE Transactions on Robotics and Automation (Dec. 2000) vol. 16, No. 6, pp. 890-898.

(Continued)

**RODNEY
BROOKS
CHANGES
PARADIG
M**

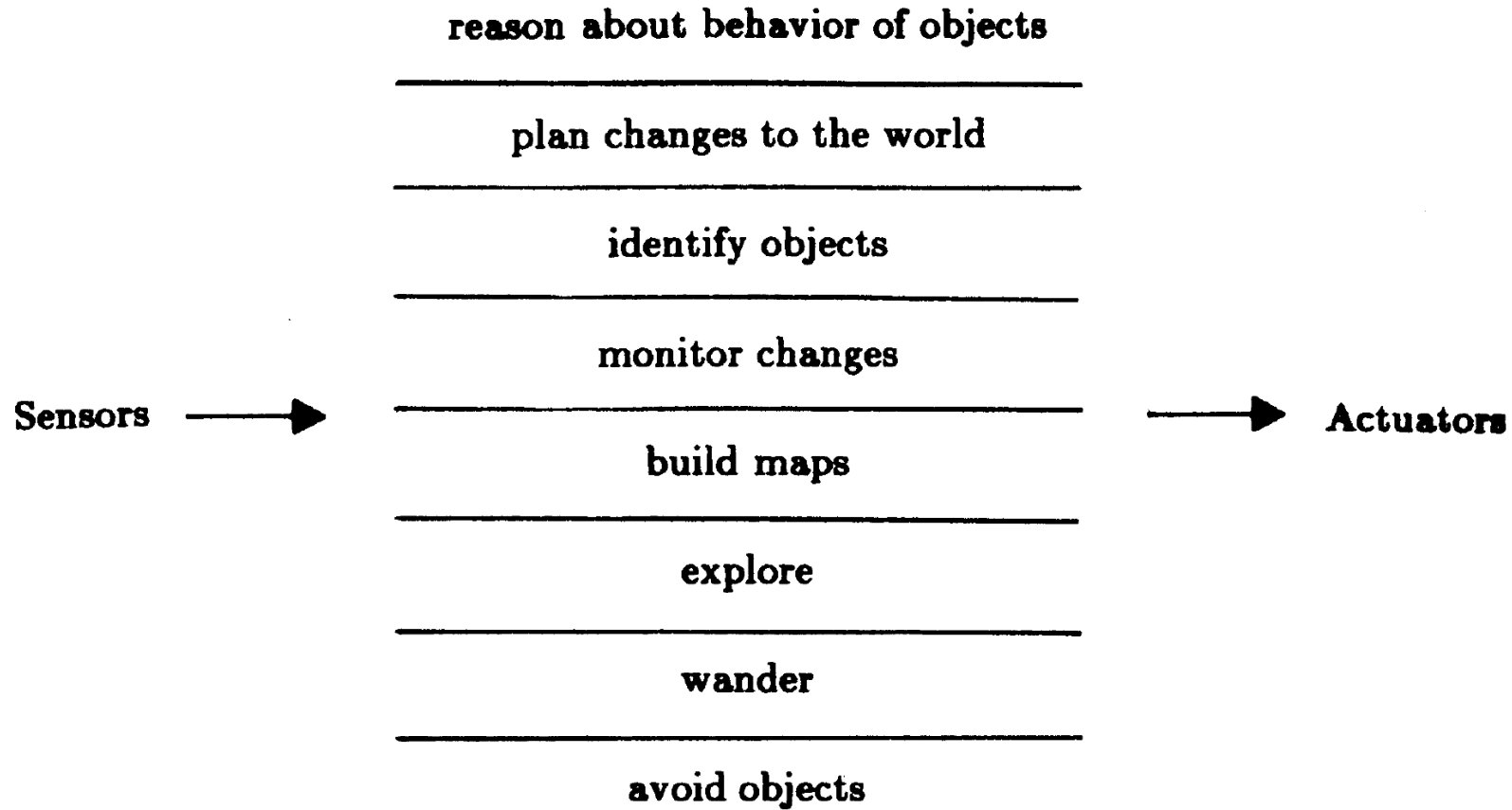
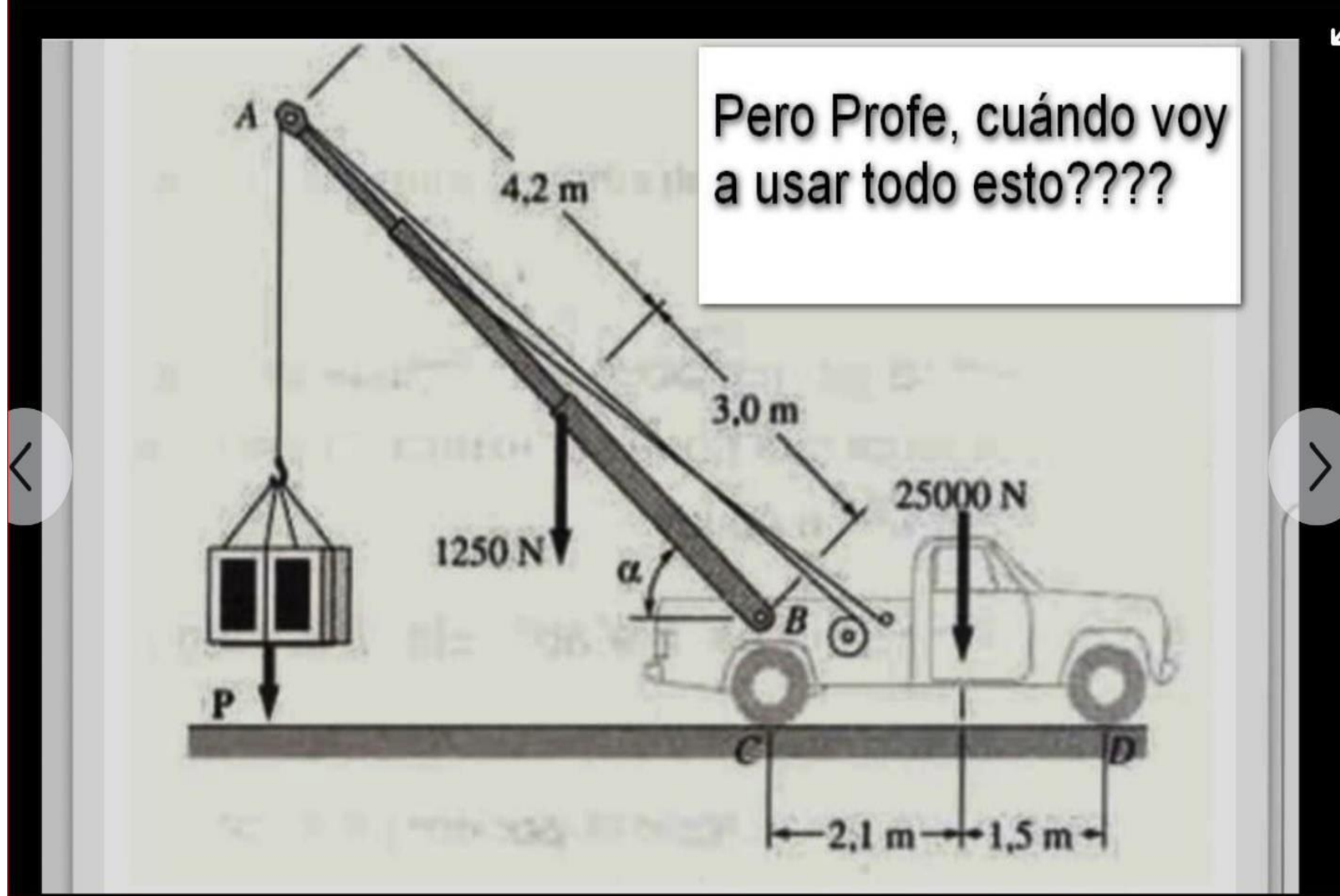


Figure 2. A decomposition of a mobile robot control system based on task achieving behaviors.

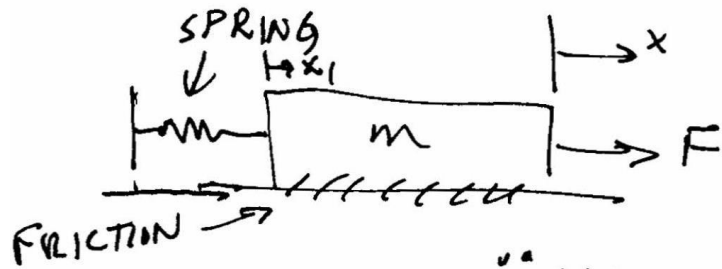
Feb 1.



Why Learn Physics - Prof - When would we use this stuff??



SHOULD HAVE LISTENED TO THE PHYSICS PROF!



M CONSTANT

$$F(t) = m \ddot{x}(t)$$

NEWTON

$$M = \frac{W}{g} \left(\frac{\text{lbs}}{\text{ft/s}^2} \right)$$

$$F = k x_1(t)$$

LINEAR SPRING

FRICTIONS (PLUS)

a) STATIC (STARTING OR STICKION)

$$F_s(t) = \pm F_s \Big|_{\dot{x}(t)=0}; \quad \vec{F} = \mu_s \vec{N}$$

b) COULOMB FRICTION

$$F(t) = F_c \text{sgn}(\dot{x})$$



OPPOSES CONSTANT VELOCITY MOTION

c) VISCOUS FRICTION

$$F(t) = B \dot{x}(t)$$

DEPENDS ON VELOCITY

$$m \ddot{x}(t) + \beta \dot{x}(t) + kx(t) = f(t)$$

[HARMAN P221

WE EXPECT

$$J \ddot{\theta}(t) + B \dot{\theta}(t) + K \theta(t) = T_{\text{Applied}}$$

$$J = \iiint_{\text{Volume}} \rho r^2 dV$$

IN GENERAL $J = c MR^2$
 $0 < c < 1.$

The Roles of Energy and Work are

Similar
LINEAR

$$E_k = \frac{1}{2} m v^2$$

$$E_p = \frac{1}{2} k x^2$$

ROTATION

$$E_k = \frac{1}{2} I \omega^2$$

$$E_p = \frac{1}{2} k \theta^2$$

$$\text{Work} = \int F(s) ds$$

$$= F s$$

If F is constant
in direction s

$$\text{Power} = \frac{dW}{dt}$$

$$P(t) = F(t) v(t)$$

$$W = T \theta$$

(Δ Kinetic Energy)

+ Δ Potential Energy

Eq 3.2.36

P 118

WATTS \rightarrow $\frac{\text{Kilowatts} \cdot \text{Hours}}{\text{time}}$

$$P(t) = T(t) \omega(t)$$



Scalars and Vectors

Glenn
Research
Center

A **scalar quantity** has only **magnitude**.

A **vector quantity** has both **magnitude** and **direction**.

Scalar Quantities

length, area, volume
speed
mass, density
pressure
temperature
energy, entropy
work, power



Vector Quantities

displacement, direction
velocity
acceleration
momentum
force
lift, drag, thrust
weight

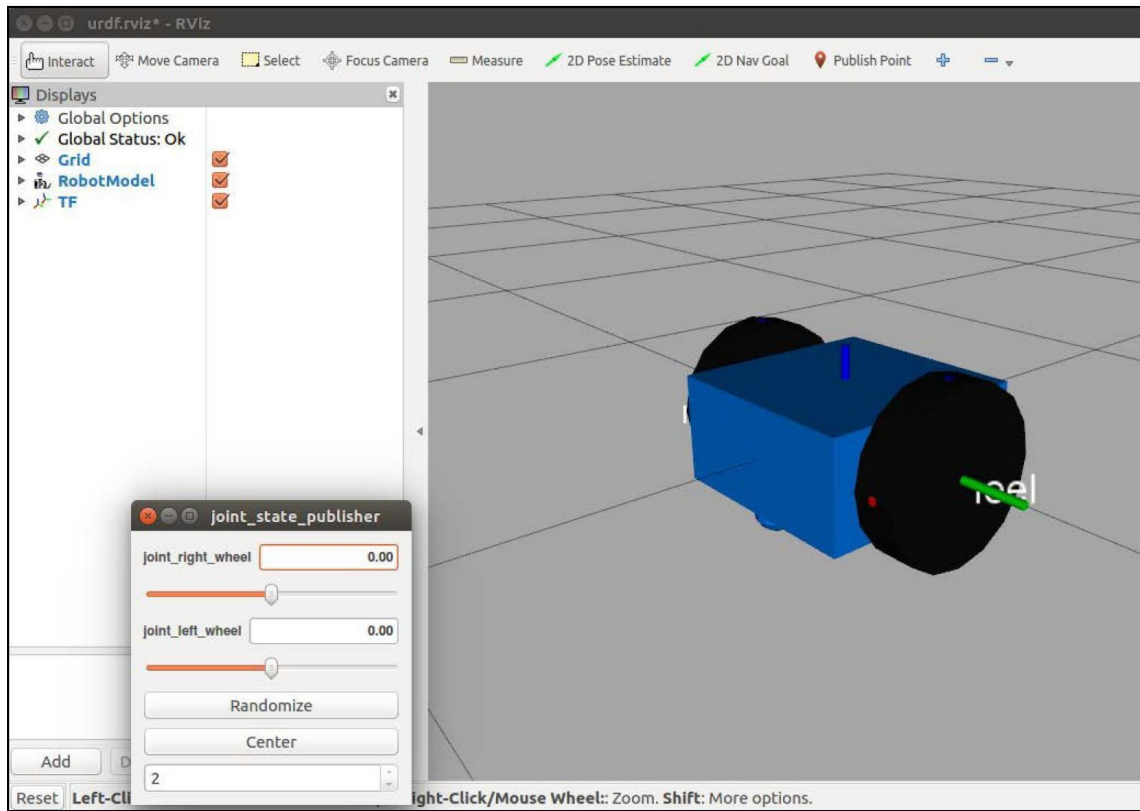


[Inertia](#)

[Stopping Distances](#)

[MotorSelection Videos](#)

Inertia and URDF Chapter 2 in Textbook (5435 Web – Text)



dd_robot5.urdf in rviz

PAGE 58 IN TEXTBOOK URDF FOR DD ROBOT

```

<?xml version='1.0'?>
<robot name="dd_robot">

  <!-- Base Link -->
  ...
  <inertial>
    <mass value="5"/>
    <inertia ixx="0.13" ixy="0.0" ixz="0.0"
              iyy="0.21" iyz="0.0" izz="0.13"/>
  </inertial>

  <!-- Caster -->
  ...
  <inertial>
    <mass value="0.5"/>
    <inertia ixx="0.0001" ixy="0.0" ixz="0.0"
              iyy="0.0001" iyz="0.0" izz="0.0001"/>
  </inertial>
</link>

  <!-- Right Wheel -->
  ...
  <inertial>
    <mass value="0.5"/>
    <inertia ixx="0.01" ixy="0.0" ixz="0.0"
              iyy="0.005" iyz="0.0" izz="0.005"/>
  </inertial>
  ...
  <!-- Left Wheel -->
  ...
  <inertial>
    <mass value="0.5"/>
    <inertia ixx="0.01" ixy="0.0" ixz="0.0"
              iyy="0.005" iyz="0.0" izz="0.005"/>
  </inertial>
  ...
</robot>

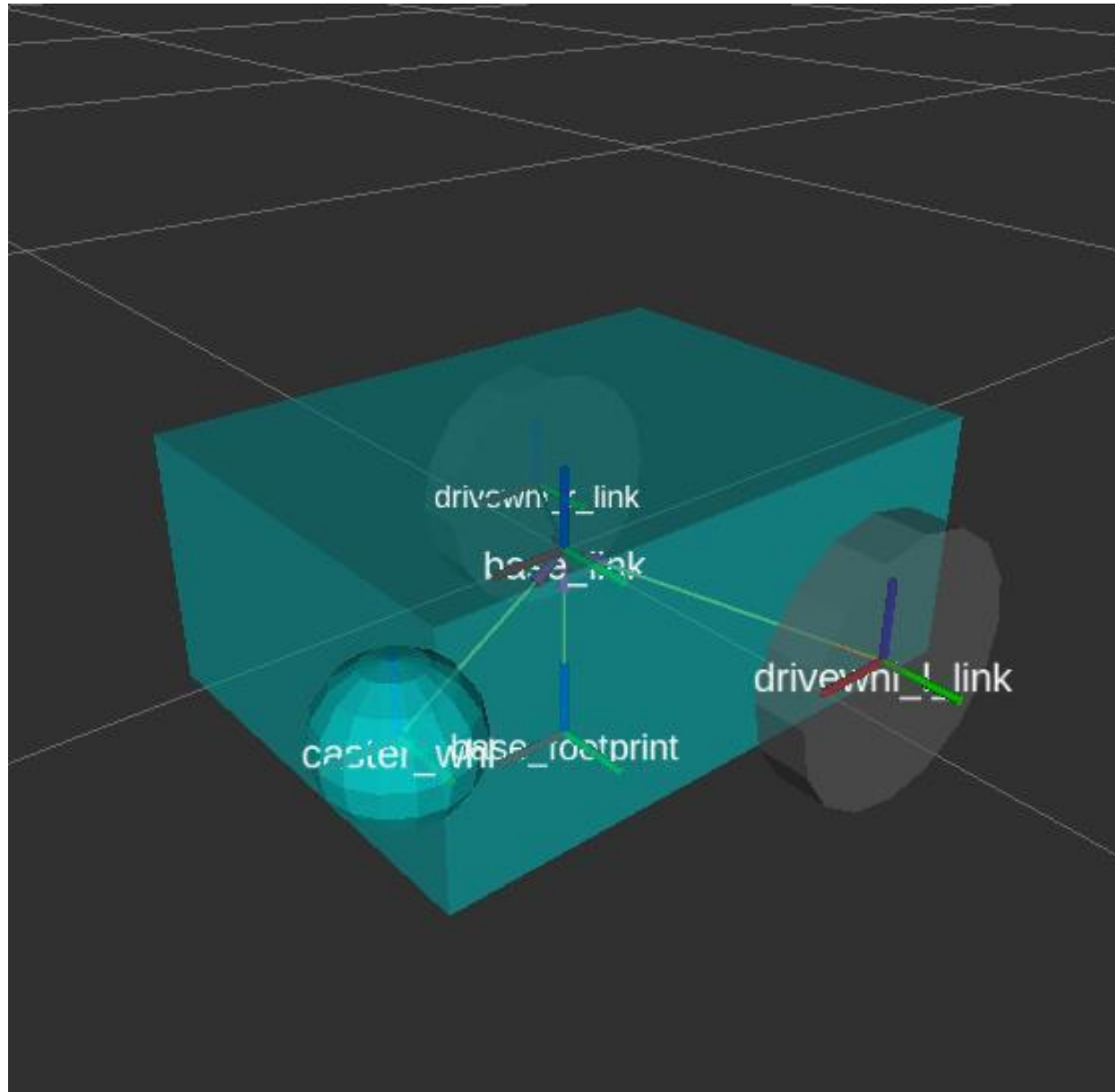
```

https://navigation.ros.org/setup_guides/urdf/setup_urdf.html

Setting Up The URDF¹

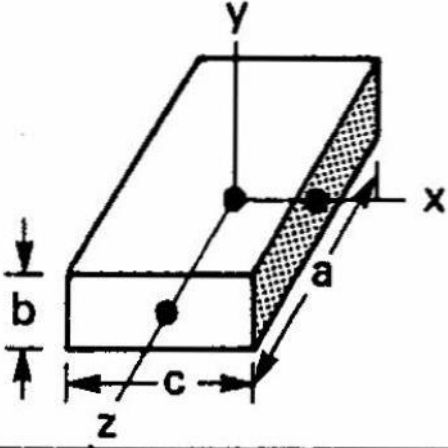
For this guide, we will be creating the Unified Robot Description Format (URDF) file for a simple differential drive robot to give you hands-on experience on working with URDF. We will also setup the robot state publisher and visualize our model in RVIZ.

Lastly, we will be adding some kinematic properties to our robot URDF to prepare it for simulation purposes. These steps are necessary to represent all the sensor, hardware, and robot transforms of your robot for use in navigation.



```
<!-- Define robot constants -->  
  <xacro:property name="base_width"  
value="0.31"/>  
  <xacro:property name="base_length"  
value="0.42"/>  
  <xacro:property name="base_height"  
value="0.18"/>  
  
  <xacro:property name="wheel_radius"  
value="0.10"/>  
  <xacro:property name="wheel_width"  
value="0.04"/>  
  <xacro:property name="wheel_ygap"  
value="0.025"/>  
  <xacro:property name="wheel_zoff"  
value="0.05"/>  
  <xacro:property name="wheel_xoff"  
value="0.12"/>  
  
  <xacro:property name="caster_xoff"  
value="0.14"/>
```

https://navigation.ros.org/_images/base-bot_1.png

	<p style="text-align: center;">Rectangular Parallelepiped</p>	$\frac{1}{12} M(a^2 + b^2)$	$\frac{1}{12} M(a^2 + c^2)$	$\frac{1}{12} M(b^2 + c^2)$
---	---	-----------------------------	-----------------------------	-----------------------------

```

<xacro:property name="caster_xoff" value="0.14"/>
<!-- Define inertial property macros -->
<xacro:macro name="box_inertia" params="m w h d">
  <inertial>
    <origin xyz="0 0 0" rpy="{pi/2} 0 {pi/2}"/>
    <mass value="{m}"/>
    <inertia ixx="{(m/12) * (h*h + d*d)}" ixy="0.0"
ixz="0.0" iyy="{(m/12) * (w*w + d*d)}" iyz="0.0"
izz="{(m/12) * (w*w + h*h)}"/>
  </inertial>
</xacro:macro>

```



```

<xacro:macro name="cylinder_inertia" params="m r
h">
  <inertial>
    <origin xyz="0 0 0" rpy="{pi/2} 0 0" />
    <mass value="{m}" />
    <inertia ixx="{(m/12) * (3*r*r + h*h)}" ixy
= "0" ixz = "0" iyy="{(m/12) * (3*r*r + h*h)}" iyz
= "0" izz="{(m/2) * (r*r)}" />
  </inertial>
</xacro:macro>

```

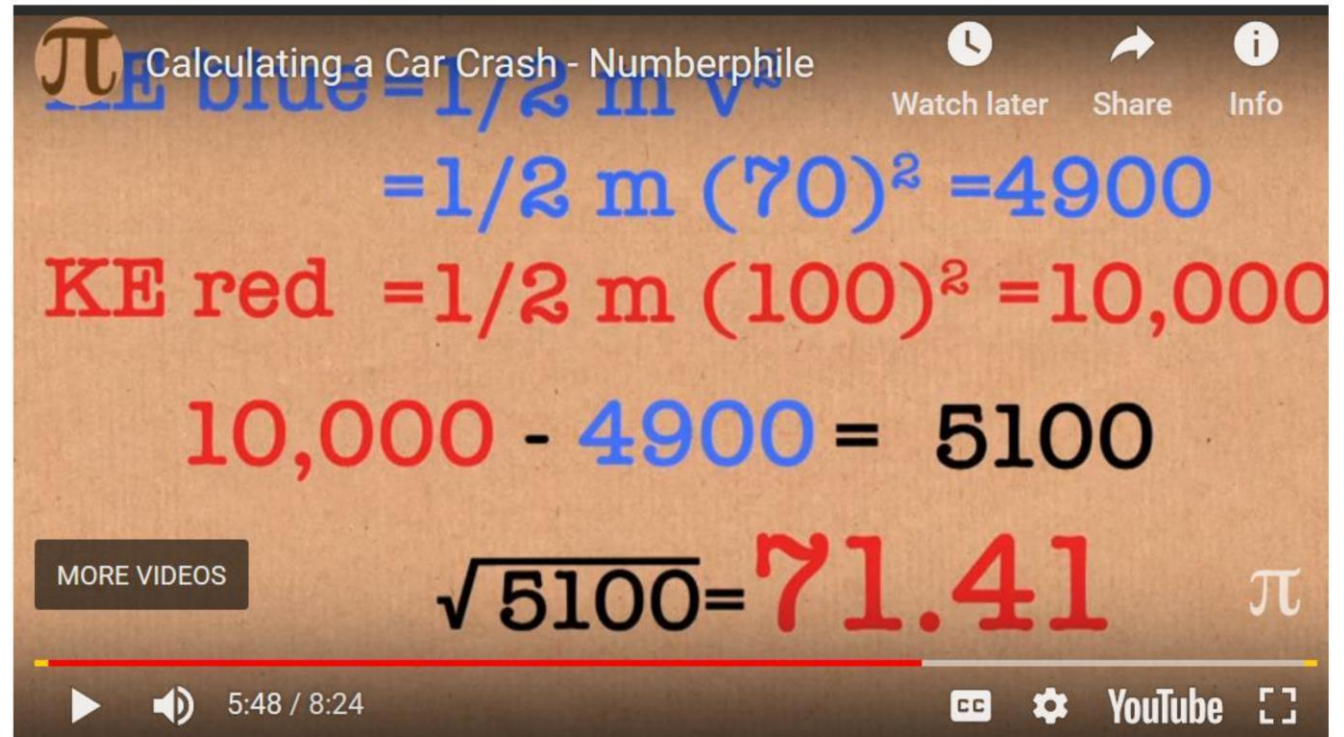
```

<xacro:macro name="sphere_inertia" params="m r">
  <inertial>
    <mass value="{m}" />
    <inertia ixx="{(2/5) * m * (r*r)}" ixy="0.0"
ixz="0.0" iyy="{(2/5) * m * (r*r)}" iyz="0.0"
izz="{(2/5) * m * (r*r)}" />
  </inertial>
</xacro:macro>

```

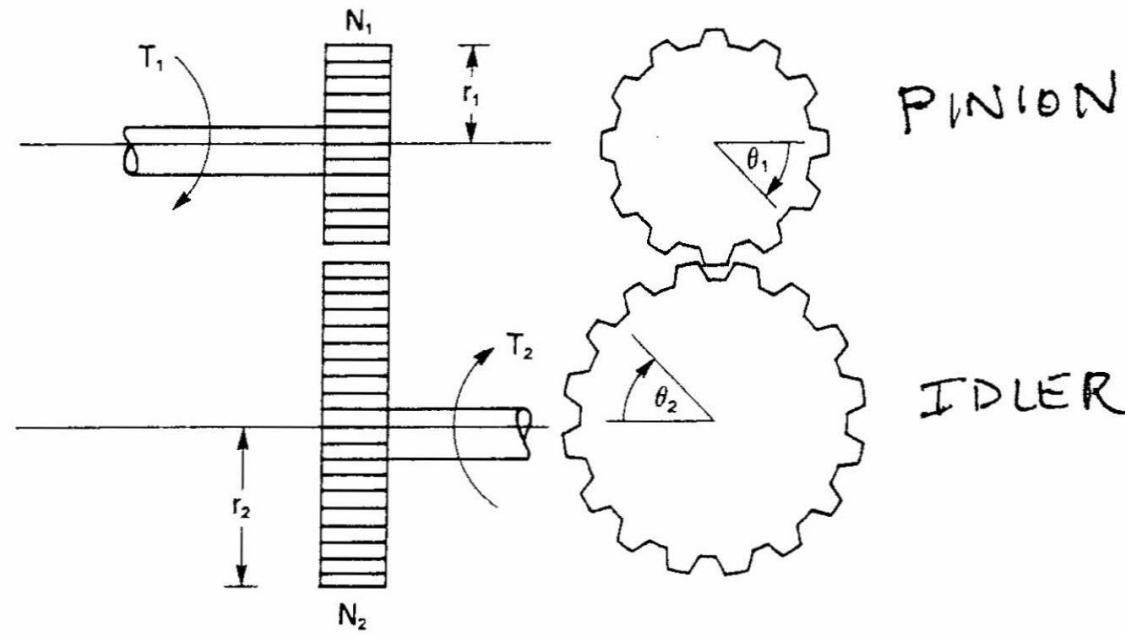


THINK AH



THINK AHEAD

Loss in KE indicates 71MPH at crash!!



Finally, noting that since the two gear radii do not vary with time, if Eqs. (3.3.2) and (3.3.3) are differentiated with respect to time, their relationship still holds but with respect to $\dot{\theta}$ (i.e., the angular velocity ω^*) or $\ddot{\theta}$ (i.e., the angular acceleration, α). Using this concept, we may write

$$\frac{N_1}{N_2} = \frac{r_1}{r_2} = \frac{T_1}{T_2} = \frac{\theta_2}{\theta_1} = \frac{\omega_2}{\omega_1} = \frac{\alpha_2}{\alpha_1} \quad (3.3.4)$$

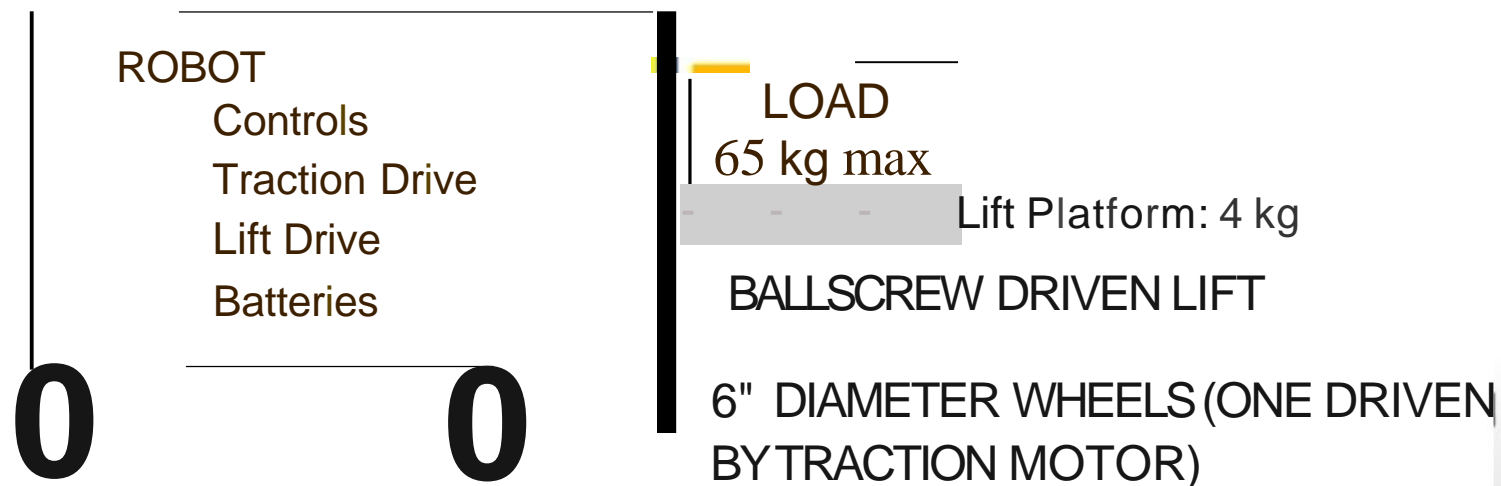
Example System

- Customer driven requirements include:
 - Max load weight (65kg)
 - Battery voltage (168Vdc)
 - Throughput, longest move time ($\leq 40s$)



Example System

- Autonomous Warehouse Fork Lift For Picking Up Pallets and Transporting Them



TOTAL WEIGHT (ROBOT AND LOAD) = 418 pounds - 190 kg mass



Control_PID_TLH

PID_EXAMPLES&Video

StateSpace MotorControl_DC_Klafter

FEEDBACK

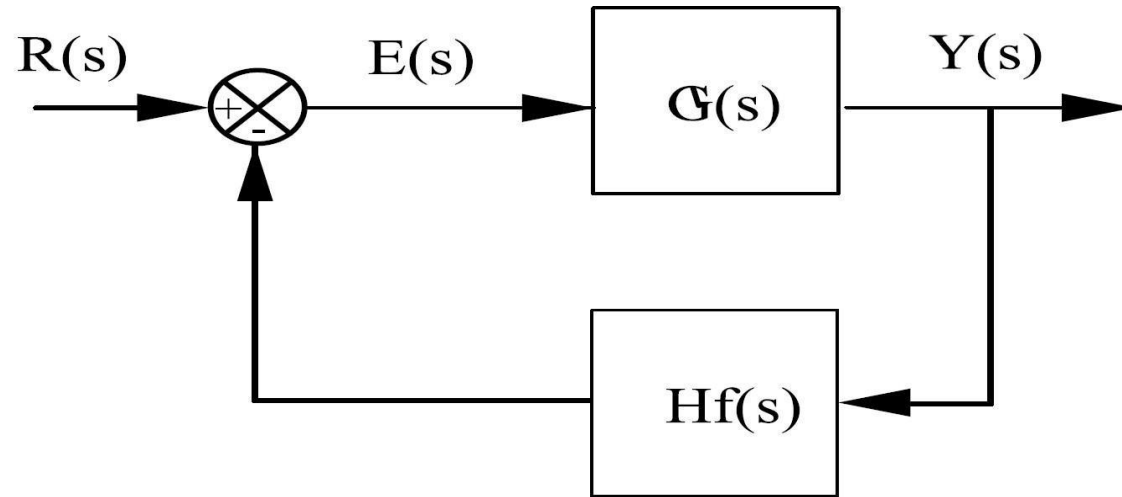


FIGURE 1.7 *Feedback control model*

If the input $R(s)$ is constant, we say the control system is a *regulator* since the object is to maintain the output at some constant value in the presence of disturbances. Temperature control or speed-control closed-loop systems are of this type. Other control systems are designed to allow the output to follow some time function as input. Such systems that control mechanical position or motion are called *servomechanisms*.

The transfer function $Y(s)/R(s)$ for the system in Figure 1.7 is derived by observing that

$$\begin{aligned} Y(s) &= G(s)E(s) \\ E(s) &= R(s) - H_f(s)Y(s) \end{aligned}$$

and eliminating $E(s)$ from these equations to yield

$G = 500$ so from Equation 1.16 in web

$$Tf_1 = \text{Gain}_1 = \frac{Y(s)}{R(s)} = \frac{500}{1 + 500(0.09)} = \frac{500}{46} = 10.87$$

5PTS

$$Tf_2 = \text{Gain}_2 = \frac{Y(s)}{R(s)} = \frac{450}{1 + 450(0.09)} = \frac{450}{41.5} = 10.84$$

5PTS

$$\Delta T_{\text{gain}} = \frac{10.87 - 10.84}{10.87} = \frac{0.030}{10.87} = 0.0028$$

OR 0.28% 5PTS

$$\Delta G(s) = \frac{500 - 450}{500} = \frac{50}{500} = 0.100$$

OR 10% 5PTS

A HW
ANSWER!!

FEEDBACK IS GREAT

FEEDBACK IS GREAT

So 10% change in $f(s)$ yields
only a 0.28% change in $T(s)$.

<https://www.electronics-tutorials.ws/systems/negative-feedback.html#:~:text=Feedback%20reduces%20the%20overall%20gain,and%20input%20and%20output%20impedances.>

Feedback **reduces the overall gain of a system** with the degree of reduction being related to the systems open-loop gain.

Negative feedback also has effects of **reducing distortion, noise, sensitivity to external changes as well as improving system bandwidth and input and output impedances.**

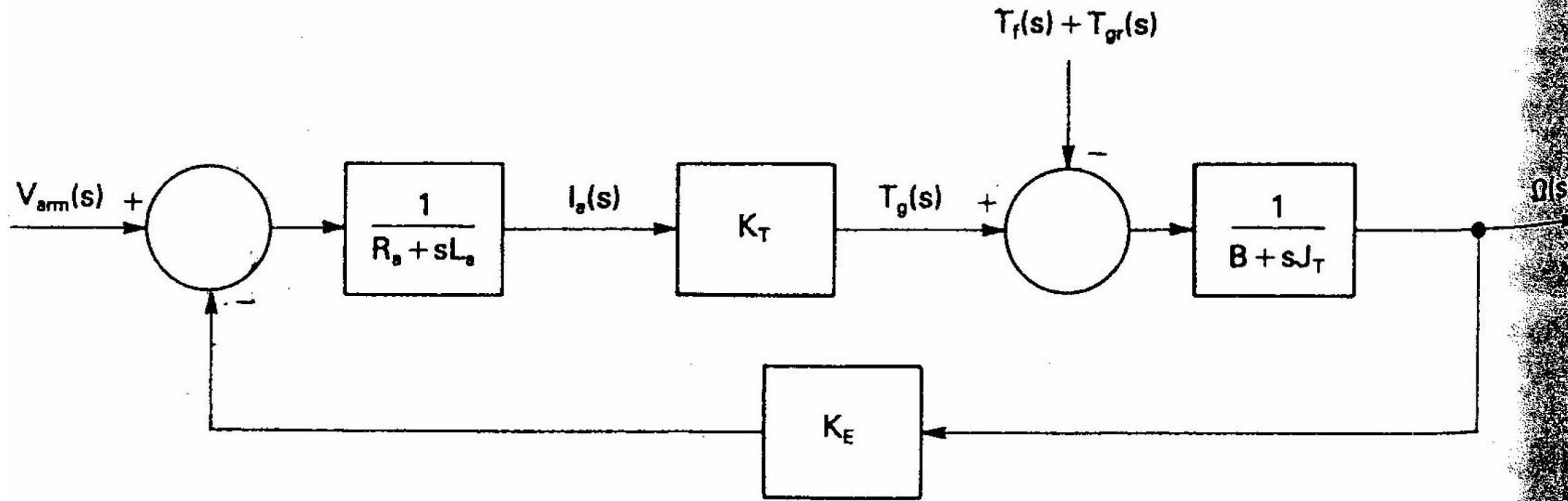


Figure 4.3.3. Block diagram of DC servomotor including gravitational and friction disturbance torques.

DC Motor

Let us find the poles [i.e., the roots of the denominator polynomial of $G_m(s)$] of a commercially available dc servomotor. For example, the parameters of an Electrocraft Corporation E530 motor are:

SE P719
OF MOTORS
PENDIX B

$$K_T = 10.02 \text{ oz-in./A} \quad \text{TORQUE CONST.}$$

$$K_E = 7.41 \text{ V/1000 rpm} \quad \text{BACK EMF}$$

$$R_a = 1.64 \text{ } \Omega \text{ (including brush resistance)}$$

$$L_a = 3.39 \text{ mH}$$

$$B = 0.1 \text{ oz-in./1000 rpm}$$

$$J_M = 0.0038 \text{ oz-in.-s}^2 \quad \text{ROTOR INERTIA}$$

Klafter

$$G_m(s) = \frac{\Omega(s)}{V_{\text{arm}}(s)}$$
$$= \frac{K_T/L_a J_T}{s^2 + [(R_a J_T + L_a B)/L_a J_T]s + (K_T K_E + R_a B)/L_a J_T}$$

$$G_m(s) = \frac{7.778 \times 10^5}{s^2 + 484.027s + 5.516 \times 10^4}$$

□ EXAMPLE 5.15 *Reduction of a Second-Order Equation*

Consider the second-order equation

$$m\ddot{x}(t) + b\dot{x}(t) + kx(t) = f(t).$$

Using the principles just defined, set $x_1(t) = x(t)$ and $x_2(t) = \dot{x}(t)$, so the first-order system becomes

$$\dot{x}_1(t) = x_2(t),$$

$$\dot{x}_2(t) = -\frac{k}{m}x_1(t) - \frac{b}{m}x_2(t) + \frac{f(t)}{m}.$$

The matrix equation is

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \frac{f(t)}{m}.$$

$$\text{For } \ddot{x} + 16\dot{x} + 64x = 0(t) \quad \vec{x}(t) = \begin{cases} x_1(t) \\ x_2(t) \end{cases}$$

$$\dot{\vec{x}}(t) = \begin{bmatrix} 0 & 1 \\ -64 & -16 \end{bmatrix} \vec{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} 0(t)$$

output is $x(t)$

$$x(t) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

$$m \ddot{x} = F_e - F_w - F_h$$

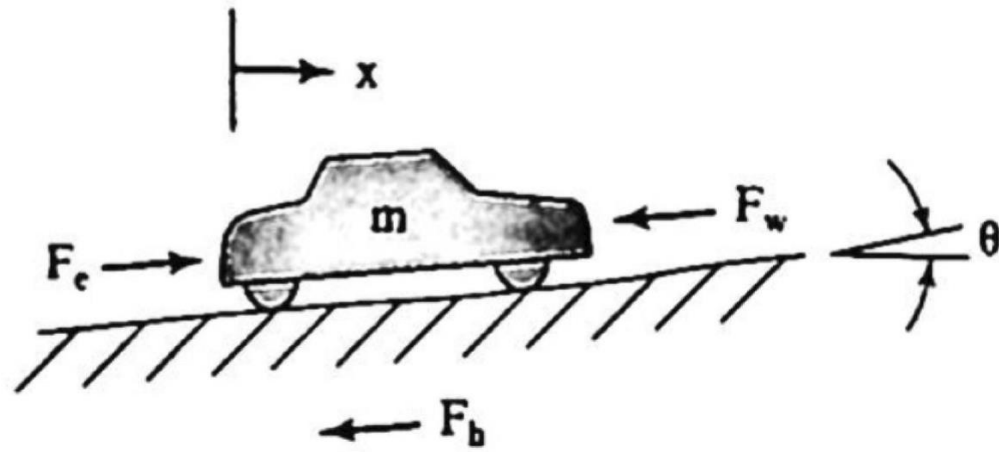


FIGURE 5.20: Automobile on a hilly road

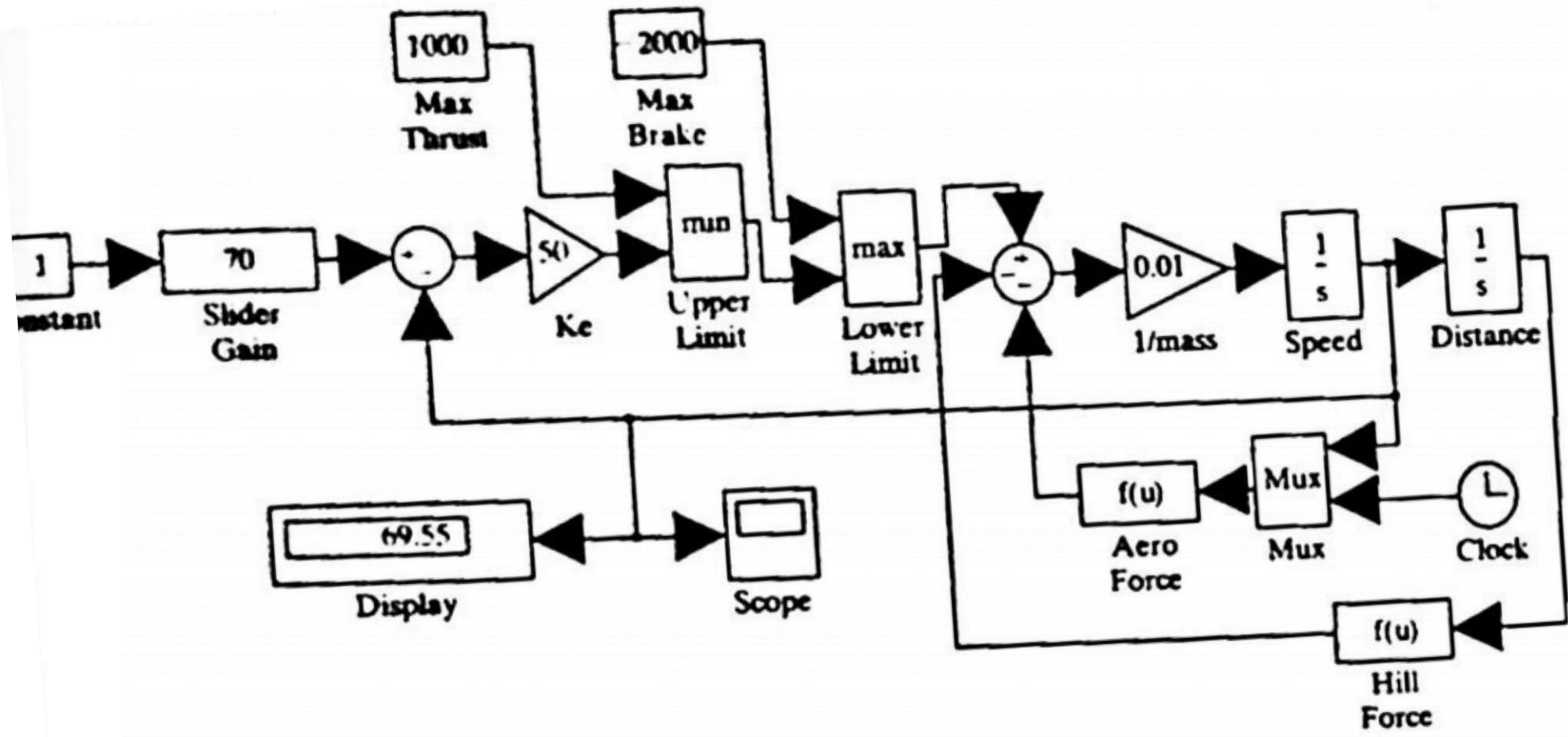


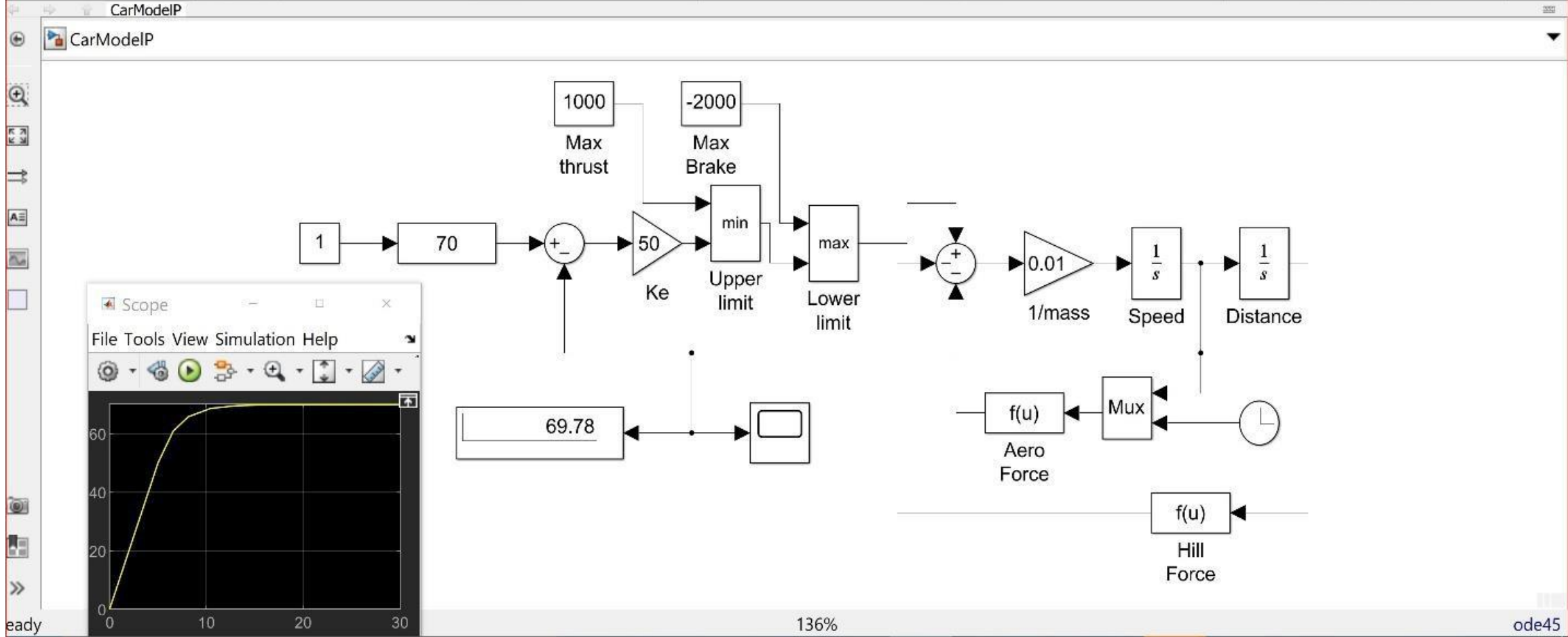
FIGURE 5.21: Automobile model with proportional speed control

CarModelIP - Simulink sponsored use

SIMULATION DEBUG MODELING FORMAT APPS

+ Open Library Browser Log Signals Add Viewer Signal Table Stop Time: 30 Normal Step Back Run Step Forward Stop Data Inspector Logic Analyzer

FILE LIBRARY PREPARE SIMULATE REVIEW RESULTS



EXAMPLE 12.6

Linearization

Suppose the acceleration of an automobile can be described by the equation of motion

$$M \frac{dv(t)}{dt} = cu(t) - \alpha v^2(t), \tag{12.33}$$

where the first term represents the acceleration caused by the engine at a throttle setting u and the second term is the drag caused by air resistance. Since this force is proportional to the square of the speed, the equation is nonlinear. Solving this by numerical techniques would not be difficult if the constants

Take (U_0, V_0) to be the “operating point,” so that the car travels at a constant speed V_0 for a constant throttle position U_0 . Inserting this condition into Equation 12.33 yields

$$M \frac{dV_0}{dt} = cU_0 - \alpha V_0^2(t) = 0,$$

or $V_0 = \sqrt{cU_0/\alpha}$. Let us assume that a small change in the throttle position leads to a small change in speed. Thus, we set

$$u(t) = U_0 + \Delta u, \quad v(t) = V_0 + \Delta v$$

and substitute again into the equation of motion. The result is

$$M \frac{d}{dt} [V_0 + \Delta v] = c[U_0 + \Delta u] - \alpha [V_0 + \Delta v]^2.$$

Using the result that $cU_0 = \alpha V_0^2$ and expanding the terms but neglecting the second-order term $-\alpha \Delta v^2$ leads to the approximation

$$M \frac{d}{dt} [\Delta v] \approx c[\Delta u] - 2\alpha V_0 \Delta v.$$

Writing $\Delta v = v_a(t)$ and $\Delta u = u_a(t)$ to indicate the linearized variables, the original differential equation of motion described by Equation 12.33 becomes the linear differential equation

$$M \frac{dv_a(t)}{dt} + 2\alpha V_0 v_a(t) = c u_a(t). \quad (12.34)$$

**LINEAR MODEL –
GOOD OVER A
LIMITED RANGE**

TWO-DIMENSIONAL TAYLOR SERIES

The notion of sequences and series of functions of a single variable as described in Chapter 6 can be extended to functions of several variables. For example, the power series expansion for a function of two variables $F(x, y)$ is

$$F(x, y) = \sum_{n=0}^{\infty} f_n(x, y) = f_0(x, y) + f_1(x, y) + \cdots + f_n(x, y) + \cdots, \quad (12.35)$$

with the terms

$$f_n(x, y) = c_{n,0} x^n + c_{n,1} x^{n-1}y + \cdots + c_{n,n-1} xy^{n-1} + c_{n,n} y^n.$$

**IN MOST ROBOT PROBLEMS, WE DEAL WITH MULTIVARIATE FUNCTIONS;
i.e. POSE AS (X, Y, Φ) FOR DD-ROBOT**

Control methods for Well- behaved systems – Particularly for linear time-invariant* (LTI) system.

Although **PID control** is the most common type of industrial controller, it does have limitations.

First, PID control is generally not suitable for systems with multiple inputs and multiple outputs (**MIMO**), as the transfer functions and differential equations used to represent the system become overly complex when more than one input (or output) is involved.

Second, PID control is based on constant parameters, so its effectiveness in controlling non-linear systems is limited.

<https://www.motioncontroltips.com/what-is-state-space-control/>

An alternative control method is state space control. The key difference between PID control (aka “transfer control”) and state space control is that the state space method takes into **account the internal state of the system**, through what are referred to as “state variables.”

These state variables describe the system and its response to any given set of inputs.

PID control, on the other hand, relies on an “observer,” which estimates the internal state of the system based on **measured inputs and outputs**.

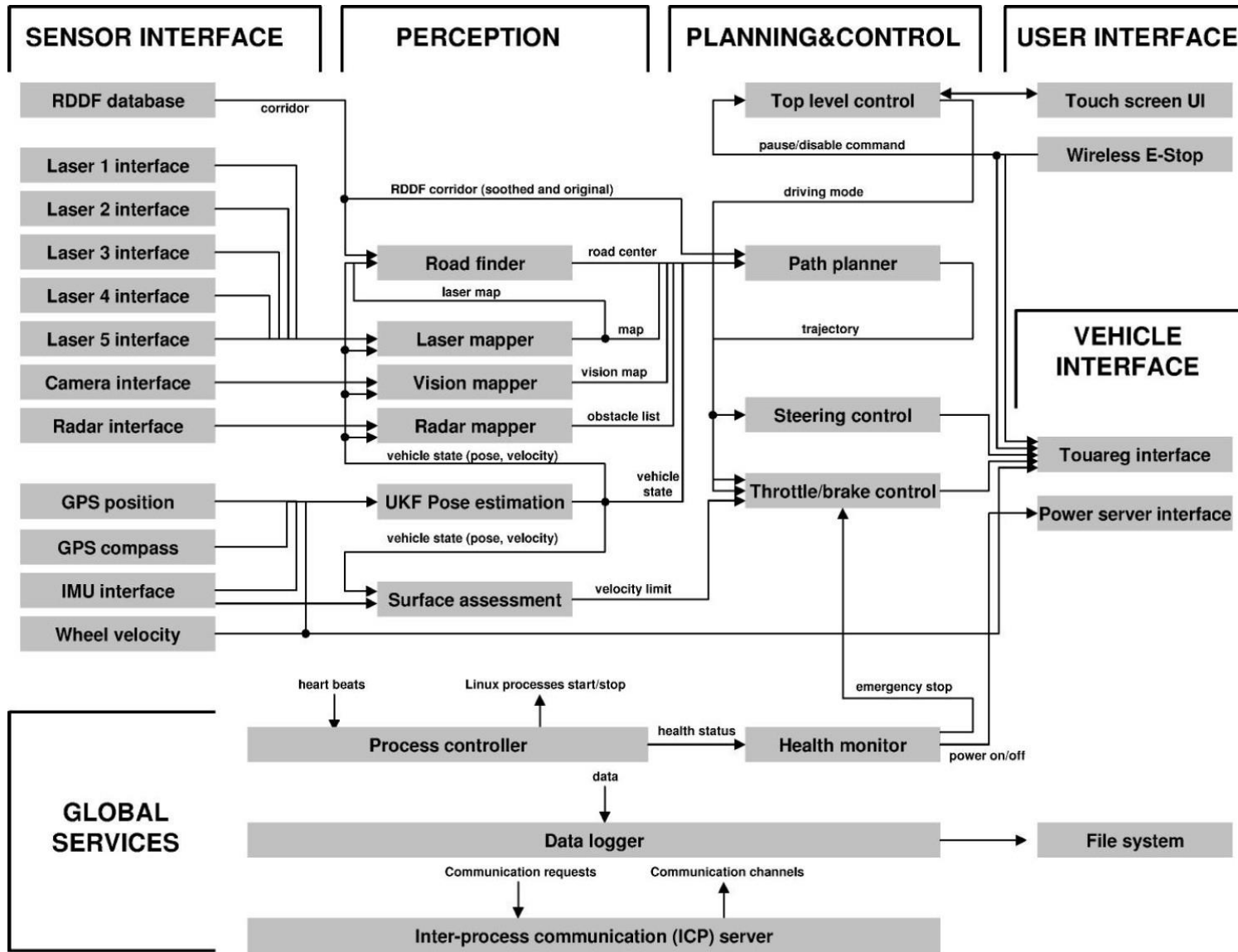


Figure 2: Software flowchart: The software is divided into six functional groups: sensor interface, perception, control, vehicle interface, user interface, and global services.