
Project Ballbot

Markus Lamprecht
Florian Müller
Michael Suffel



TECHNISCHE
UNIVERSITÄT
DARMSTADT

TU Darmstadt, RTM

Inhaltsverzeichnis

1	Item - List	2
2	Simulation	5
2.1	Launch	5
2.2	Simulation design	5
2.3	Gazebo Parameters	6
2.4	Control	6
2.4.1	Plugins	6
2.4.2	Launch	6
2.5	Sensors	7
2.5.1	IMU	7
3	Model	8
3.1	Composition	8
3.2	Assumptions	8
3.3	Model Parameters	9



1 Item - List


Item	#	W.[g]	Weblink	Picture
OpenCR Board (Controlling the motors, IMU)	1	60	github_wiki	
UpBoard (Main PC)	1	96	127€	
Intel RealSense R200	1	9.4	datasheet, 84.15€	
Laser Distance Sensor	1	124	specs, 100€	
Battery: LI-PO 11.1 1800mAh LB-12 19	1	132	44.90€	
Turtlebot3 Layers(125cmx125cm)	4			
XM430-W350-R Dynamixel (Motors)	3	82	robotis,250€	
Ball(alum., dia.: 140mm, material thickness 2.5mm)	1	400	ball-tech gmbh,40€.	
Omni wheels(dia: 60mm, thickness:25mm)	3	51.46	10.38€	
Kreisring (PLA, 3D printeted)	1	28		
Halterung (PLA, 3D printeted)	3	18		
Mitnehmer (PLA, 3D printeted)	3	8		
Plain washer (Beilagscheibe),(PLA, 3D printeted)	3	0.45		
M3 (Mutter-Halterung-Kreisring-Layer)	9			
M2.5 (Kreisring-Layer)	2			
M3x8mm Halterung	6		Zylinderkopf (Imbus)	
M3x22mm Layer	3	1.34	Zylinderkopf (Imbus)	
M2.5x22 (Motoren-Halterung)	12		Sechskant	
M2.5x38 (Motoren-Rad)	3		Zylinderkopf (Imbus)	
M2.5x24 (Layer)	2		Zylinderkopf (Imbus)	
M2x6mm (Mitnehmer-Motor)	12		Zylinderkopf (Imbus)	
Distanzbolzen	???		???	
Total Cost: 1176€ + Cost of opencr board and all plastic (incl. tb3 structure) and scrwes				

Tabelle 1.1: My caption

Type	Size	Amount	Place
Cylinderhead screw	M3 x 11mm	8	Motor mounts
Cylinderhead screw	M2,5 x 22mm	16	Motor plate
Cylinderhead screw	M2 x 6 mm	18	Wheel shaft
Cylinderhead screw	M2,5 x 36 mm (38 mm)	5	Wheel shaft cover
Cylinderhead screw	M3 x 20 mm (21mm)	4	Layer mounting
Nut	M2	5	Layer mounting
Cylinderhead screw	M2,5 x 22mm (23mm)	4	Layer mounting

TODO:

1. Abmessungen von einer struckture layer
2. upboard1-link noch eintragen

2 Simulation

TODO: check if controller works
check why imu fails

2.1 Launch

These files are executed one after another:

1. bb_simulation: ballbot.launch
2. bb_description: bb_description.launch
3. bb_description -> urdf: bb.xacro
4. bb_description -> urdf: bb.urdf.xacro
5. bb_description -> urdf: common_properties.xacro
6. bb_description -> urdf: bb.gazebo.xacro

2.2 Simulation design

Ballbot SDF Reference: [Ballbotmodel](#)

We use not the sdf but the xacro description as in this example [here](#).

```
rrbot_control
├── CMakeLists.txt
├── config
│   └── rrbot_control.yaml
├── launch
│   ├── rrbot_control.launch
│   ├── rrbot_rqt.launch
│   └── rrbot_rqt.perspective
└── package.xml
rrbot_description
├── CMakeLists.txt
├── launch
│   ├── rrbot.rviz
│   └── rrbot_rviz.launch
├── meshes
│   └── hokuyo.dae
├── package.xml
├── urdf
│   ├── materials.xacro
│   ├── rrbot.gazebo
│   ├── rrbot.xacro
│   └── rrbot.xml
└── rrbot_gazebo
    ├── CMakeLists.txt
    ├── launch
    │   └── rrbot_world.launch
    ├── package.xml
    ├── worlds
    │   └── rrbot.world
```

Gazebo uses different physics engines:

- Open Dynamics Engine (ODE) (Default)
- Bullet
- Dynamic Animation and Robotics Toolkit (DART)
- Simbody

which all have different friction etc. models.

Files:

- `bb.urdf.xacro`: Link's: Visual description of the Robot and its collision model(STL file). Pose Mass and Inertias. Joint's: Pose,axis,effort and velocity limits, friction.
- `common_properties.xacro`: Macros for color definition.
- `bb.gazebo.xacro`: gazebo references dynamics of the links: friction parameters (`mu1,mu2`),

Gazebo Parameter's List:

name(xacro)	description	value	sdf group
<code>mu1</code>	is the Coulomb friction coefficient for the first friction direction	1.0	ode
<code>mu2</code>	is the friction coefficient for the second friction direction (perpendicular to the first friction direction)	2.0	ode
<code>kp</code>	spring constant equivalents of a contact as a function of <code>SurfaceParams::cfm</code> and <code>SurfaceParams::erp</code>		ode
<code>kd</code>	spring damping constant equivalents of a contact as a function of <code>SurfaceParams::cfm</code> and <code>SurfaceParams::erp</code> .		ode
<code>cfm</code>	Constraint Force Mixing parameter.		ode
<code>erp</code>	Error Reduction Parameter.		ode
<code>min_depth</code>	Minimum depth before ERP takes effect.		ode
<code>max_Vel</code>	Maximum interpenetration error correction velocity. If set to 0, two objects interpenetrating each other will not be pushed apart.		ode
<code>slip1</code>	Artificial contact slip in the primary friction direction		ode
<code>slip2</code>	Artificial contact slip in the secondary friction direction.		ode

See: [ODESurfaceParams](#)

2.3 Gazebo Parameters

2.4 Control

sobald diff drive plugin angeschaltet drehen sich die raeder viel zu schnell

Diff Drive in `ballbot.launch` an oder ausschalten.

in `bb.gazebo.xacro` transmission und controller festlegen.

zudem yaml file(currently I use: `effort_controllers/JointVelocityController`)

Effort Joint Interface as Hardware Interface is used.

Do this example first: http://gazebosim.org/tutorials/?tut=ros_control

Also try this bb8 gazebo tutorial: <https://www.youtube.com/watch?v=j5qC91448p8>

2.4.1 Plugins

- `gazebo-ros-control`
- `diff drive`

2.4.2 Launch

`roslaunch rrbot_control rrbot_control.launch`

These files are executed one after another:

1. `load config`
2. `controller_spawner`

2.5 Sensors

2.5.1 IMU

We want to simulate the IMU of the opencr board. STRG+T to see imu topic values! [Imu of opencr board simulated](#)

Simulate like this: rviz rviz dann als fixed frame nimm: imu_link. Und add topic imu und waehle als topic ballbot/sensor/imu

The simulated IMU outputs values like: orientation (x,y,z,w), angluar velocity(x,y,z), linear velocity(x,y,z), linear acceleration(x,y,z).

The opencr real IMU gives values like: orientation(x,y,z,w), angular velocity(x,y,z), linear acceleration(x,y,z) see http://turtlebot3.readthedocs.io/en/latest/appendix_opencr.html

3 Model

3.1 Composition

The Ballbot consists of three parts, which are depicted in Figure 3.1.

- Body with motors
- 3 omni-directional wheels
- Ball

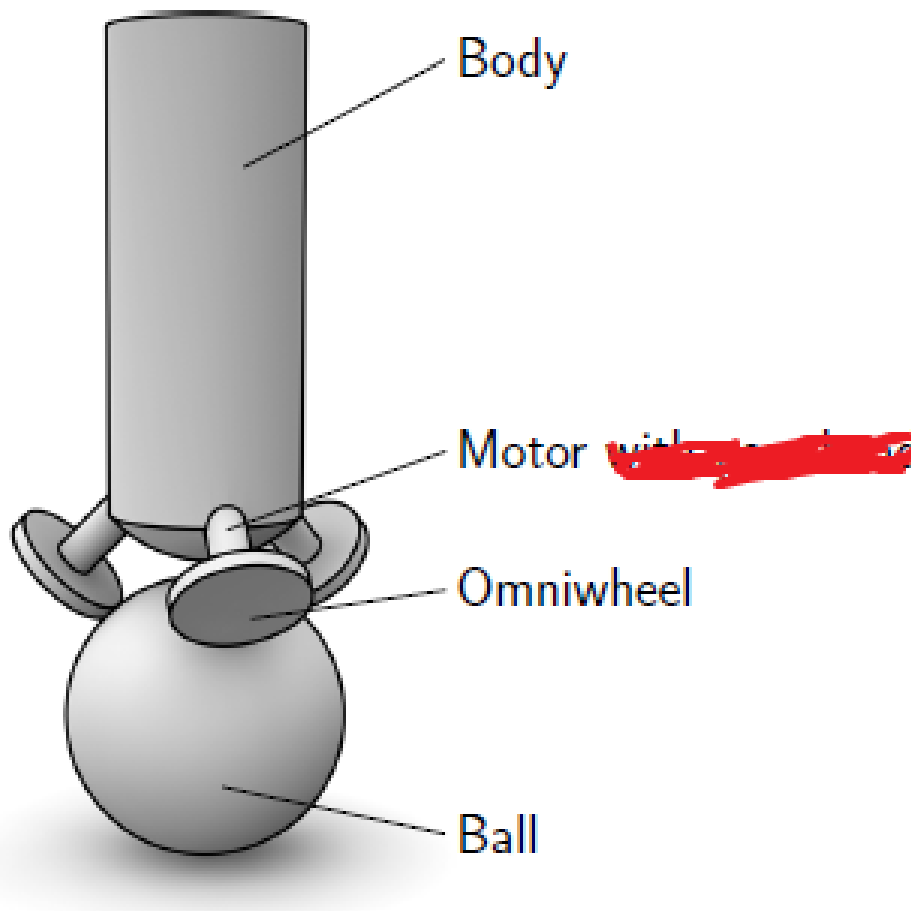


Abbildung 3.1: Parts for the 3D-Model

3.2 Assumptions

To reduce the complexity of the system, the following assumptions are made:

Tabelle 3.1: My caption

Parameter	Variable	Value	Source
Mass of the ball	m_K	0,4 kg	Datasheet
Mass of the body	m_B	1,646 kg	SolidEdge
Mass of the virtual wheel	m_{VW}	0,384 kg	Measured
Radius of the ball	r_K	0,07 m	Datasheet
Radius of the body	r_B	0,0703 m	Measured
Radius of the Wheels	r_W	0,03 m	Datasheet
Height of the center of gravity	l	0,24045 m	SolidEdge
Height of the body	h	0,34294 m	SolidEdge
Inertia of the Ball	Θ_K	0,00131 kgm ²	Computed
Inertia of the Body (x-axis)	Θ_{Bx}	0,08751 kgm ²	SolidEdge
Inertia of the Body (y-axis)	Θ_{By}	0,08788 kgm ²	SolidEdge
Inertia of the body (z-axis)	Θ_{Bz}	0,00329 kgm ²	SolidEdge
Inertia of the body (xy plane)	Θ_{Bxy}	-0,00001 kgm ²	SolidEdge
Inertia of the body (xz plane)	Θ_{Bxz}	0,00203 kgm ²	SolidEdge
Inertia of the body(zy plane)	Θ_{Bzy}	0,00018 kgm ²	SolidEdge
Gear ratio	i	353,5	Datasheet
Gravitational acceleration	g	9,81 m/s ²	BachelorThesis

- No slip between the contact points between the ball/ground and wheels/ball
- No friction; except the friction, which occurs at the rotation of the ball around the z-axis
- No deformation
- Fast motor dynamics; The controlling of the motor is much faster than the controller of the Ballbot
- Ball moves only horizontal

3.3 Model Parameters

Traegheitsmoment kugel Hohlzylinder: $J = m \frac{r_i^2 + r_a^2}{2} = 0.4 * \frac{65^2 + 70^2}{2} kg * 10^{-6} m^2 = 1.825 * 10^{-3} kg m^2$
 Traegheitsmoment wheel Vollzylinder: $J = m \frac{1}{2} r^2 = 0.05146 * 0.020^2 = 1.0292 * 10^{-4} kg m^2$