Skoltech
Skolkovo Institute of Science and Technology

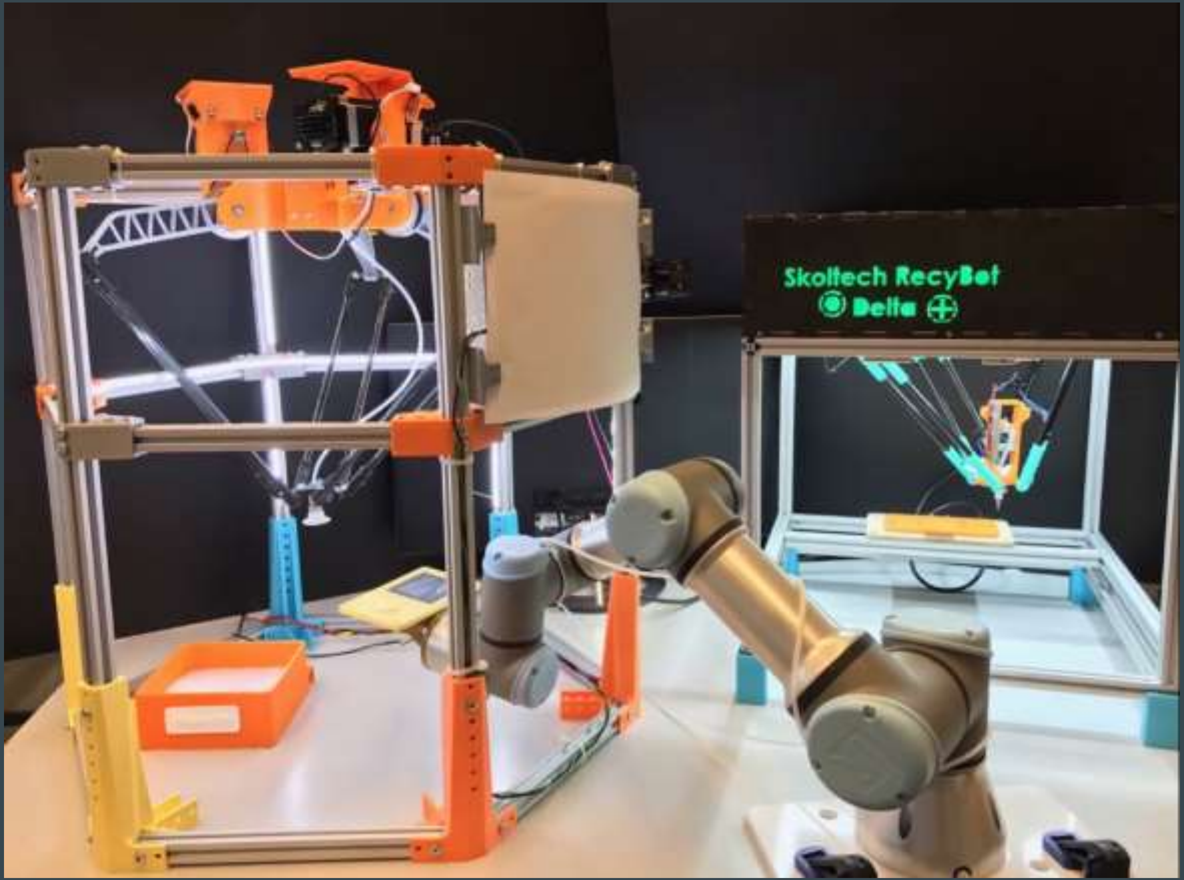
“Development of constraint based robot description method for automatic kinematics calculation of parallel manipulator”

“Разработка метода описания робота с помощью геометрических ограничений для автоматического расчета кинематики параллельного манипулятора”

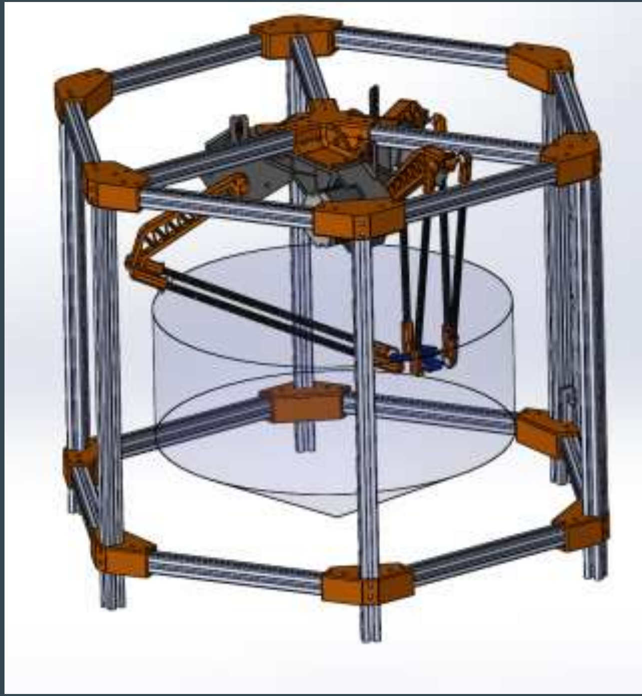
Student: Dmitry Ermachenkov
Scientific Adviser: Professor Dzmitry Tsetserukou

May, 2018

Motivation: Recybot project challenges



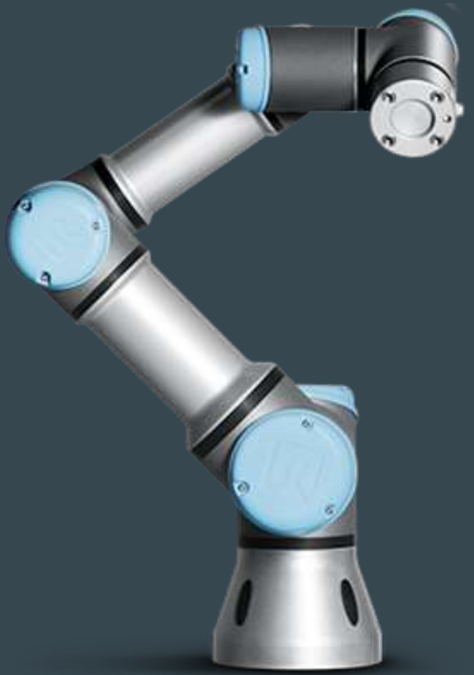
Motivation: Recybot project challenges



Recybot 2.0 Delta robot

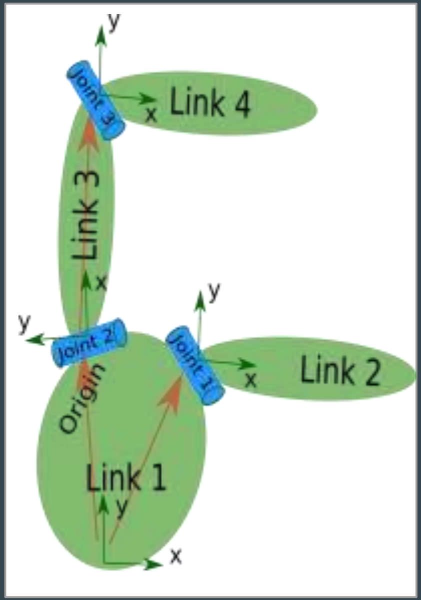
ROS

Open Source Robotics Foundation



Serial UR robot

Motivation: ROS flaws



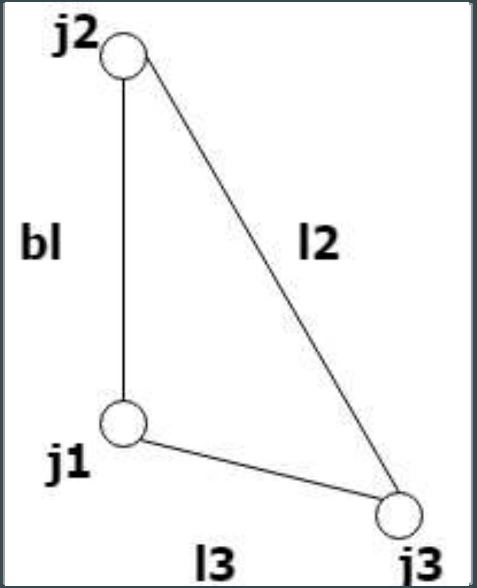
Tree mechanism[1]

```
<robot name="test_robot">
  <link name="link1" />
  <link name="link2" />
  <link name="link3" />
  <link name="link4" />

  <joint name="joint1" type="continuous">
    <parent link="link1"/>
    <child link="link2"/>
  </joint>
  <joint name="joint2" type="continuous">
    <parent link="link1"/>
    <child link="link3"/>
  </joint>
  <joint name="joint3" type="continuous">
    <parent link="link3"/>
    <child link="link4"/>
  </joint>
</robot>
```

URDF of tree mechanism[1]

Serial robot in ROS motion planning

Motivation: ROS flaws



Close-loop mechanism

```xml
<robot name="test_robot">
  <link name="link1" />
  <link name="link2" />
  <link name="link3" />
  <joint name="joint1" type="continuous">
    <parent link="link1"/>
    <child link="link2"/>
  </joint>
  <joint name="joint2" type="continuous">
    <parent link="link1"/>
    <child link="link3"/>
  </joint>
  <joint name="joint3" type="continuous">
    <parent link="link2"/>
    <child link="link1"/>
  </joint>
</robot>
```

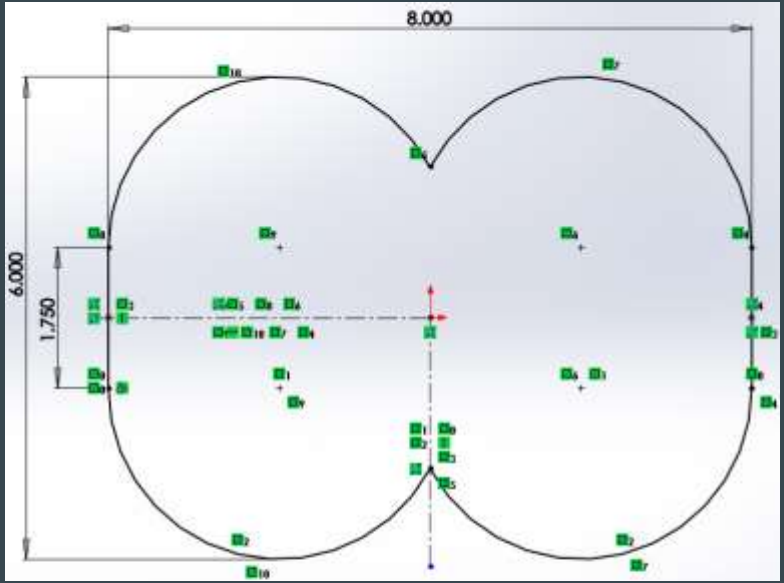
URDF of close-loop mechanism

Motion planning of close-loop robot in ROS

Aims

Develop a description method, that supports both parallel and serial robots

Create a practical instrument (ROS package) for parallel robot kinematics solution and integrate it into ROS



Solidworks geometric constraints

Objectives

- *Make literature review about prior work*
- *Propose description format*
- *Choose correct GCS for robot description*
- *Make simulation with simple mechanism and delta-robot,*
- *solved by GCS*
- *Develop ROS package for parallel robot control*
- *Test developed package on real robot*

Methods

- *Literature review*
- *Comparative analysis*
- *Simulation of robot kinematics*
- *Software prototyping and development*
- *Architecture synthesis*
- *CAD model design*
- *Hardware experiment*

Results: GCS comparison

GCS	Open-Source API	Easy to install	Docs	Constraint types #	Speed	3D support	Aim
GeoSolver	yes	no	no	small	slow	yes	CAD
KDL Orocos	yes	?	no	large	medium	yes	Simulation
ShapeOp	yes	no	no	small	slow	yes	Tissue model
pygeosolve	yes	yes	no	small	slow	no	2d sketches
SketchSolve	yes	no	no	medium	medium	no	2d sketches
V-REP GCS	yes	no	no	medium	medium	yes	Simulation
PAROSOLID	no	?	yes	large	fast	yes	CAD
C3D SOLVER	no	?	yes	large	fast	yes	CAD
SolveSpace	*yes*	*yes*	*no*	*medium*	*fast*	*yes*	*CAD*

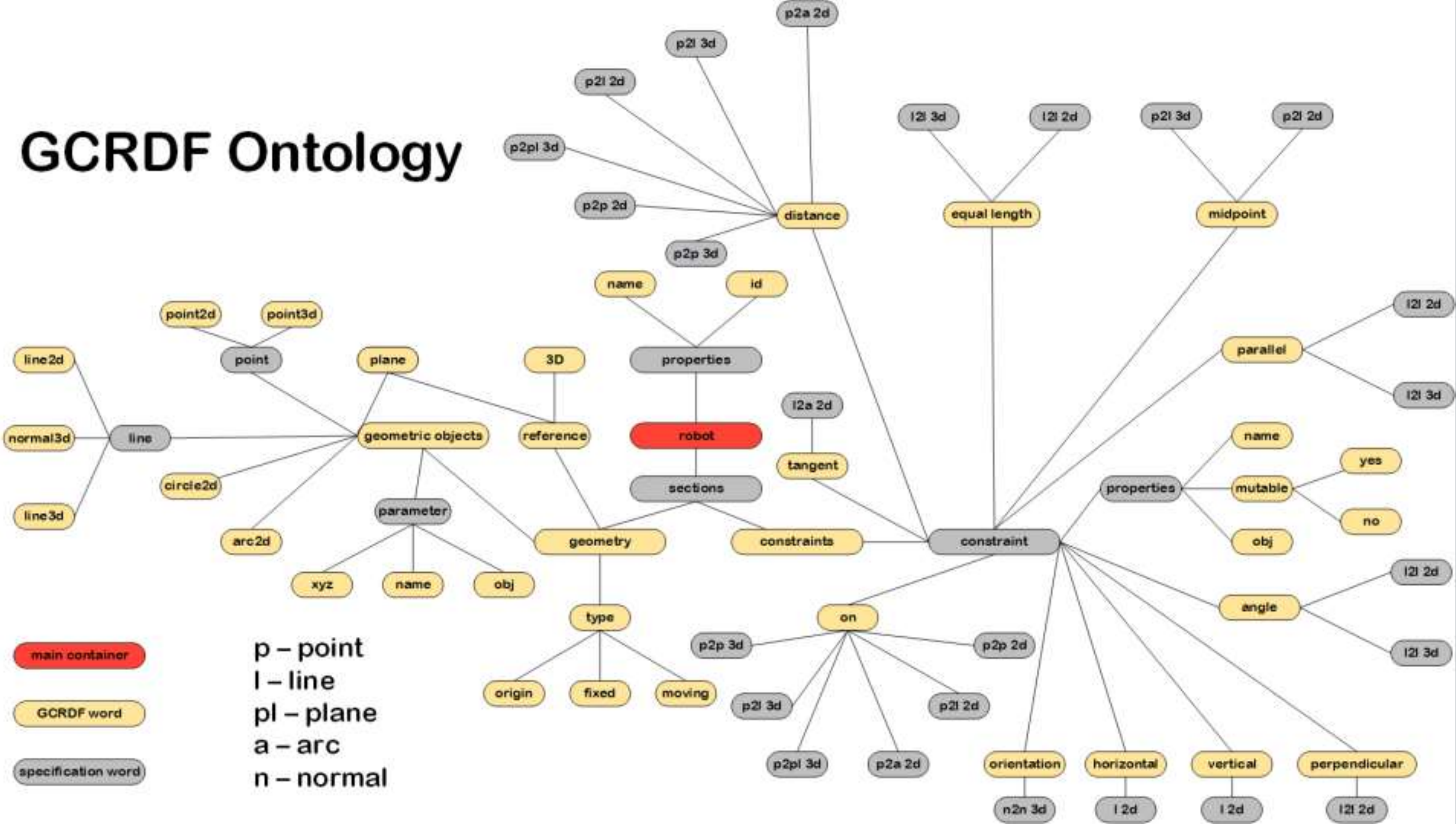
Powers:

- *Good speed for 2d*
- *Many constraints*
- *Easy to install*

Drawbacks:

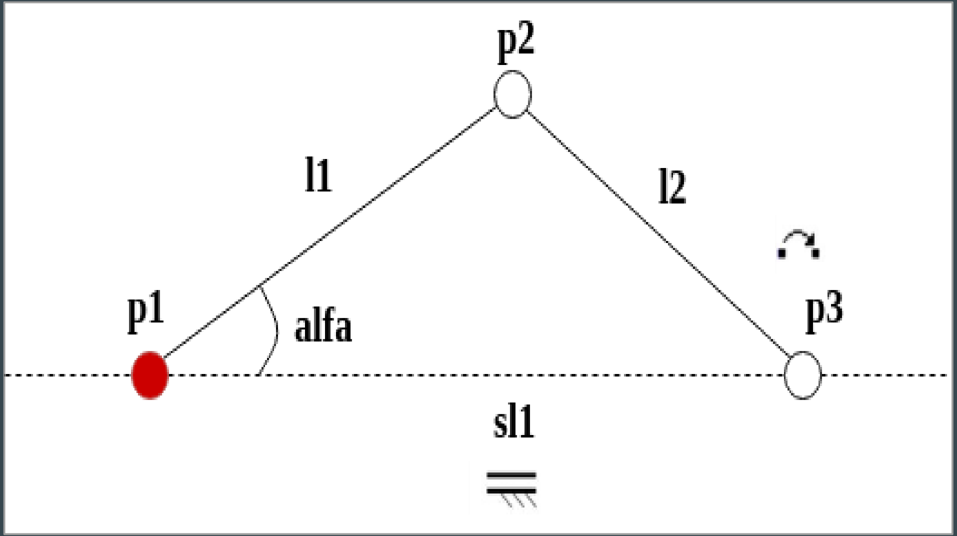
- *Small documentation*
- *Mostly for Windows*
- *Some crucial API is closed*

GCRDF Ontology

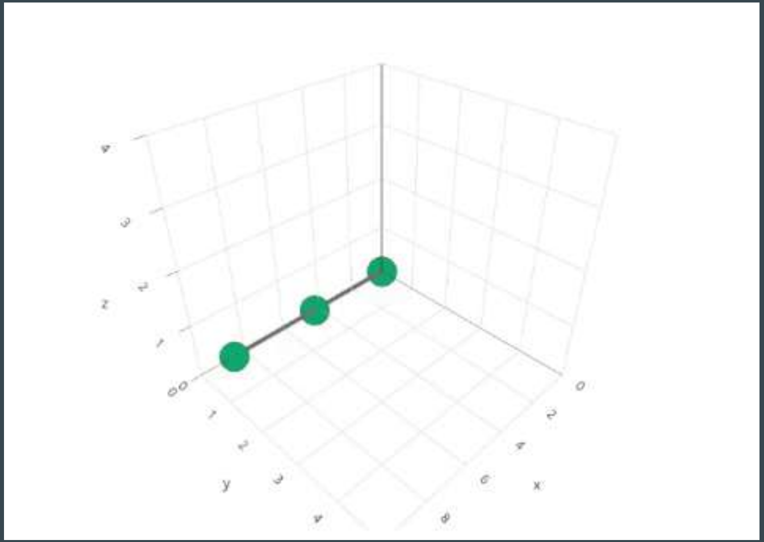


p – point
l – line
pl – plane
a – arc
n – normal

Results: Simple mechanism description



Simple mechanism description

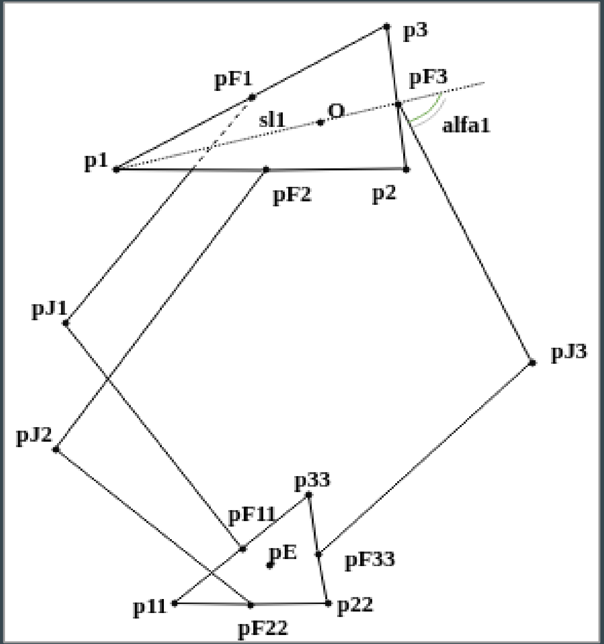


Simple mechanism simulation

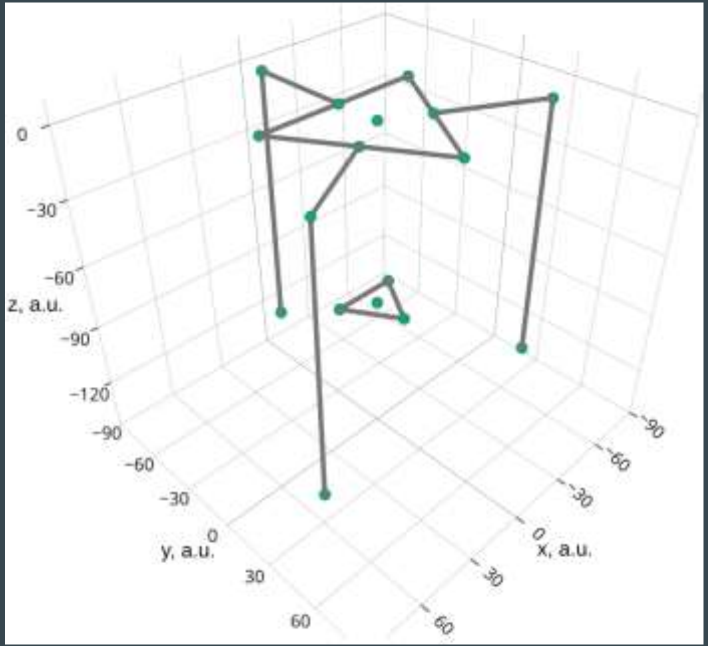
Results: Simple mechanism GCRDF file

```
<robot name="2d_serial_robot">
  <geometry type="origin">
    <point3d name="OriginP", xyz="0,0,0"/>
    <normal3d name="WNormal", qwxyz="1,0,0,0"/>
    <plane name="WPlane", origin="OriginP", normal="WNormal"/>
  </geometry>
  <geometry type="fixed", plane="WPlane">
    <point2d name="p1", xy="0,0"/>
    <point2d name="sp2", xy="20,0"/>
    <line2d name="sl1", p1="sp1", p2="sp2"/>
  </geometry>
  <geometry type="moving", plane="WPlane">
    <point2d name="p2", xy="3.0*sqrt(2),0"/>
    <point2d name="p3", xy="6.0*sqrt(2),0"/>
    <line2d name="l1", p1="p1", p2="p2"/>
    <line2d name="l2", p1="p2", p2="p3"/>
  </geometry>
  <constraints plane="WPlane">
    <on obj1="p3", obj2="sl1"/>
    <distance value="3.0*sqrt(2)", p1="p1", p2="p2"/>
    <distance value="3.0*sqrt(2)", p1="p2", p2="p3"/>
    <angle name="alfa", obj1="OA", obj2="lo", value="0.0", mutable="yes"/>
  </constraints>
</robot>
```

Results: Delta mechanism simulation

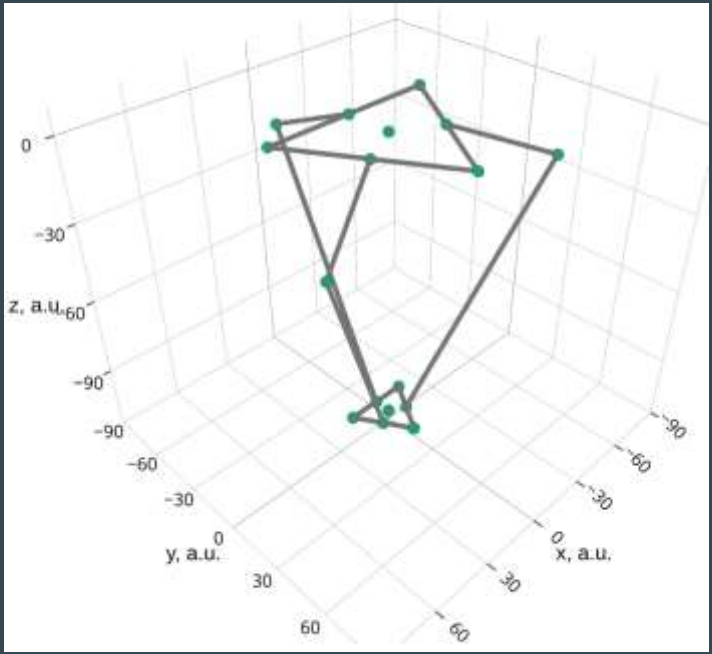


Delta mechanism structure

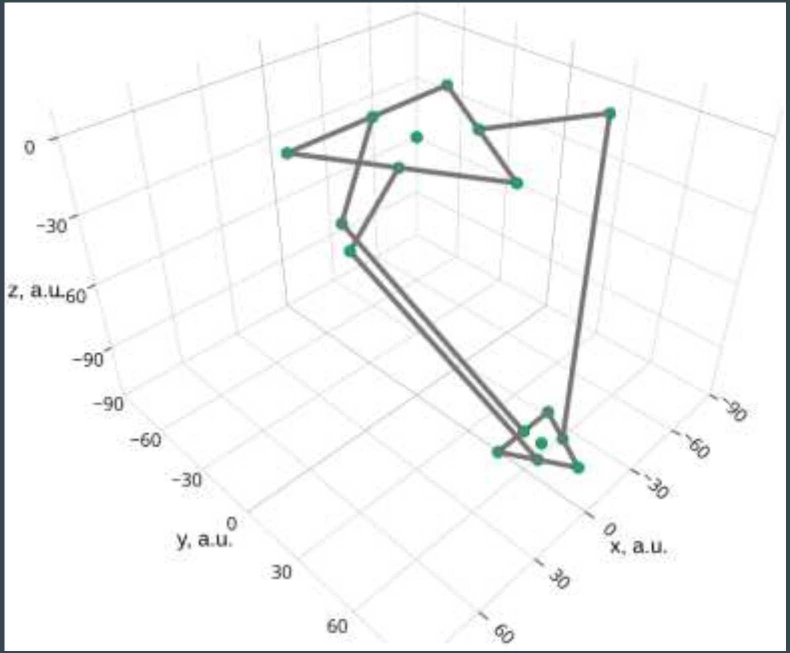


Delta mechanism simulation
Step 1

Results: Delta mechanism simulation

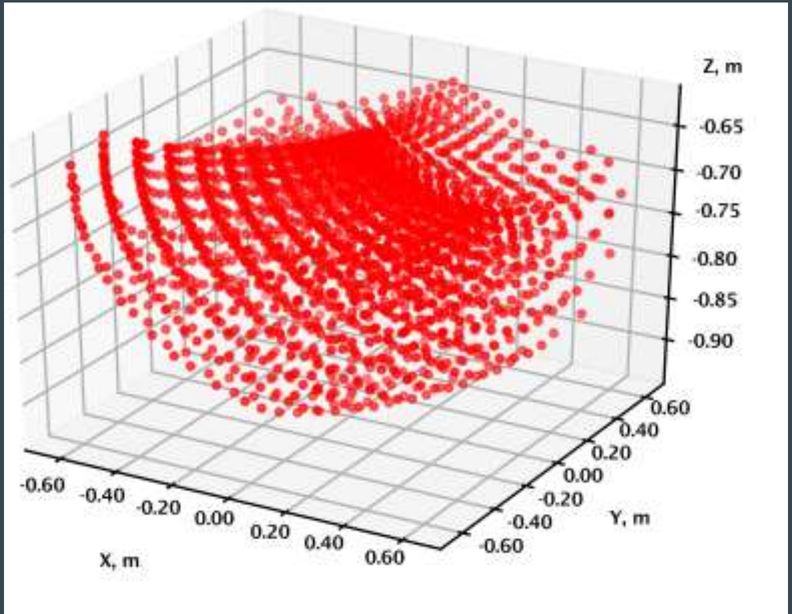


Delta mechanism simulation
Step 2

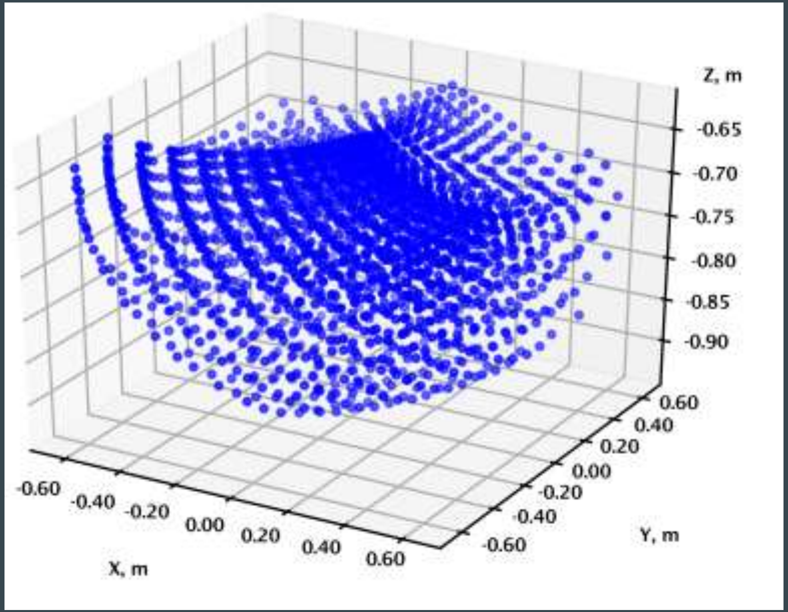


Delta mechanism simulation
Step 3

Results: Delta workspace simulation



Python workspace simulation



GCP workspace simulation

Results: GCS simulation comparison

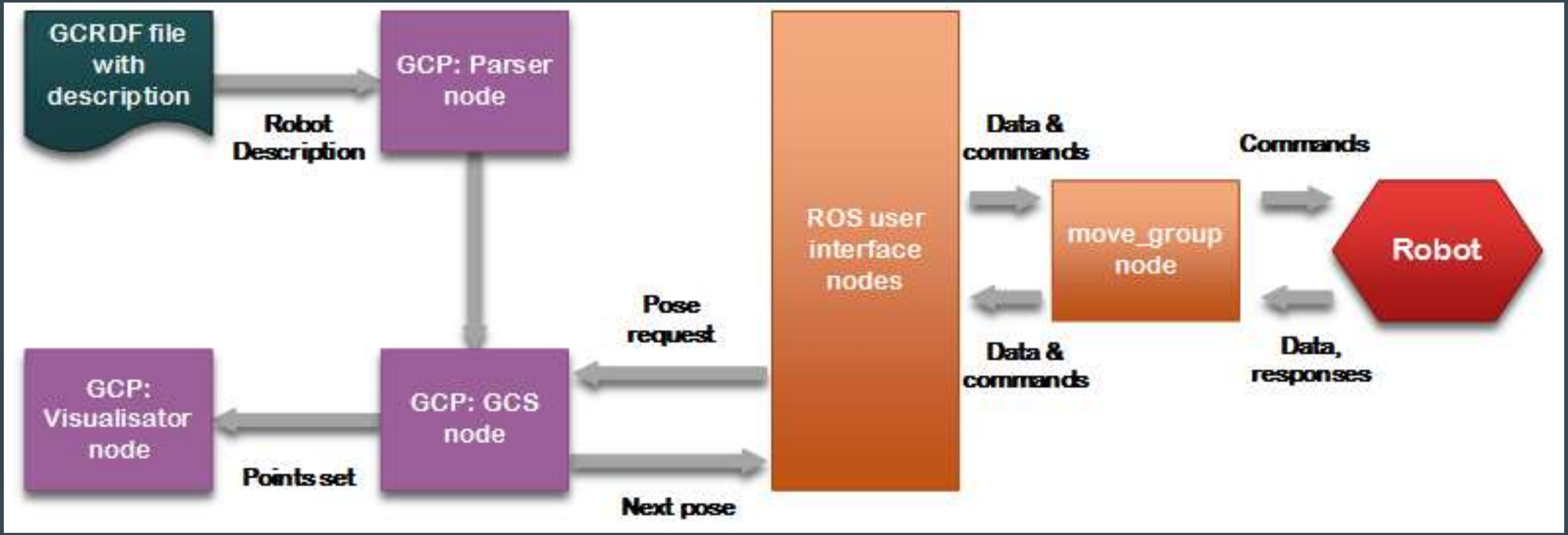
- GCS approach is slower
- GCS approach is less accurate

GCS can be applied to mechanisms of different complexity in an unified way

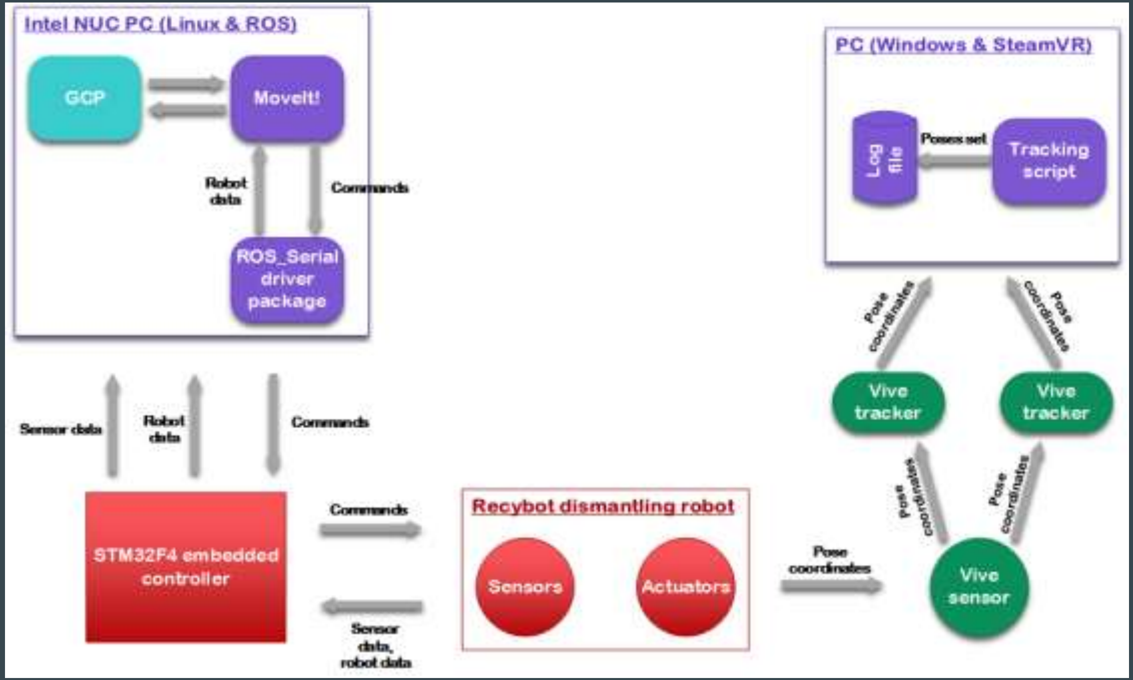
Benchmarks for serial mechanism			
	SolveSpace	Geosolver	Python
Time	0.258s	0.326s	0.00001s
Accuracy	0.104mm	0.122mm	-

Benchmarks for delta mechanism[5]			
	SolveSpace	Geosolver	Python
Time	0.361s	failed	0.014s
Accuracy	0.133mm	failed	-

GCP software architecture



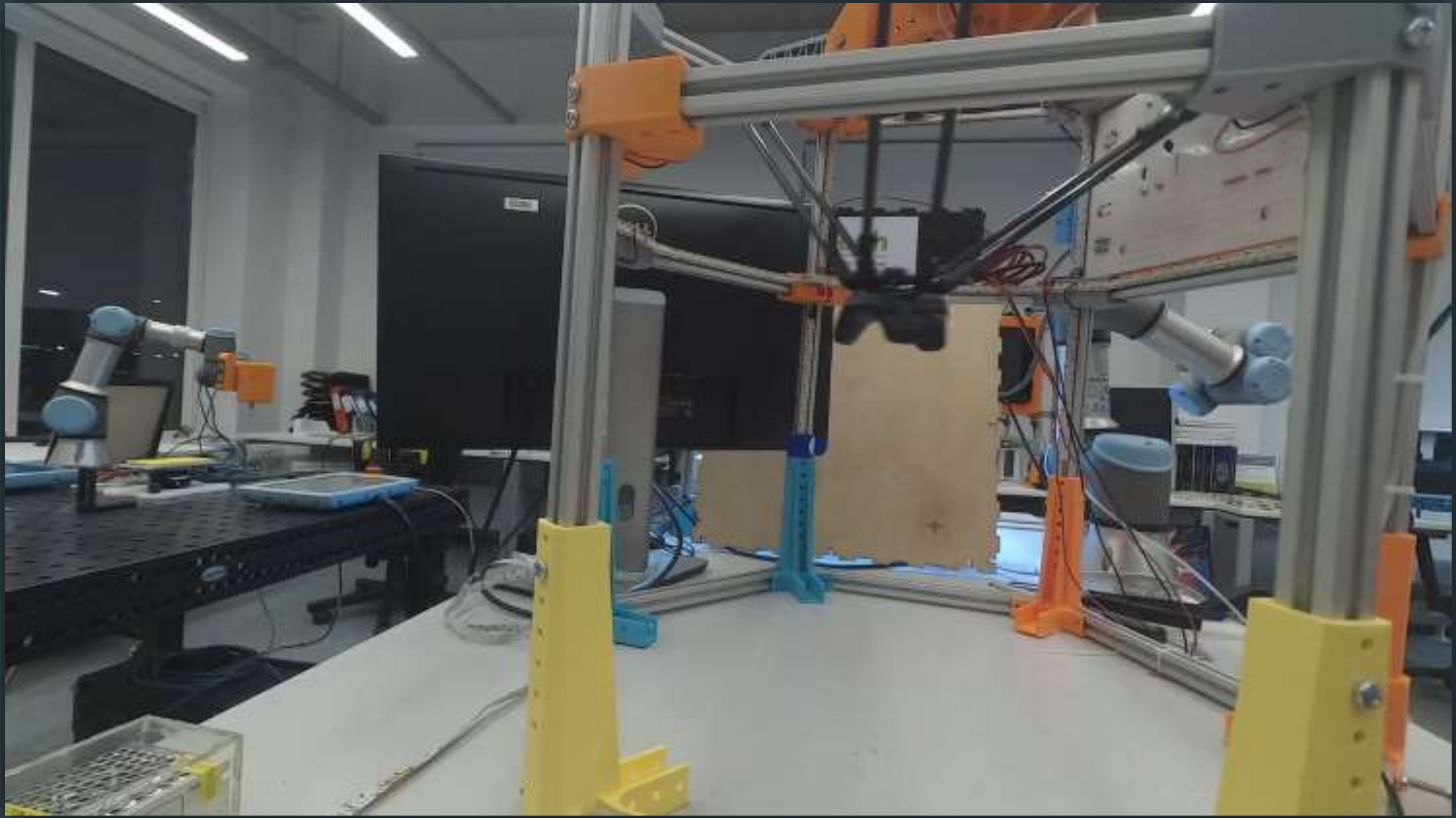
Structural scheme of experiment setup



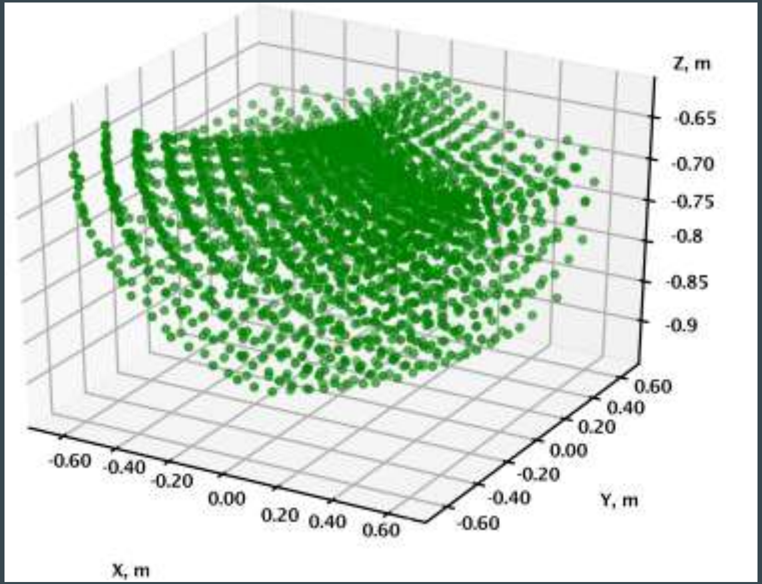
Experiment setup scheme

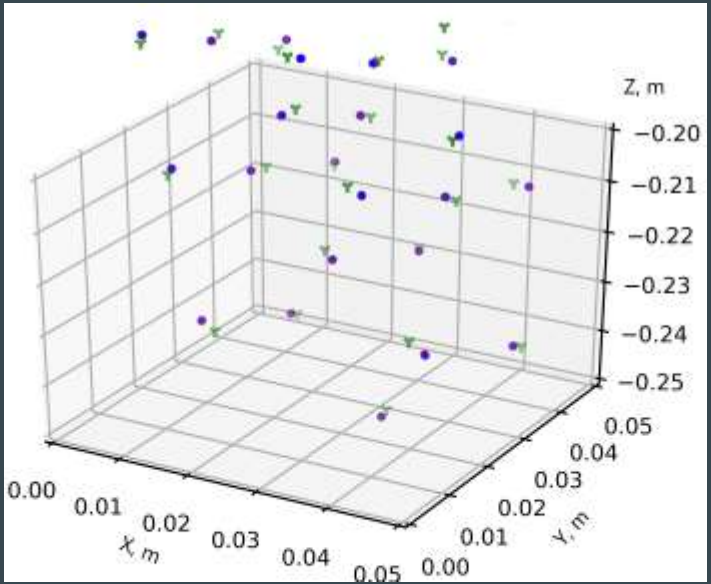
Delta robot 3d model

Experiment video



Experiment and simulation workspace





- 500 dots
- Mean experiment accuracy: 0.2 mm
- HTC Vive accuracy: submillimiter (~0.1 mm)

Application

NGP Recybot demo

- Different industrial parallel robots
- Control of robot structures with underived kinematics
- Integration into MoveIt! (part of robotic software stack used worldwide)

Conclusions & innovation

- Parallel and serial robots can be controlled in unified way
- More complex parallel structures can now be studied and used in practice

My novelty: new approach to parallel robots description, that is

- As natural as designing a mechanism in CAD
- Practical and extendable
- Applicable to other mechanism structures

Acknowledgements

Professor Dzmitry Tsetserukou

(Skoltech ISR)

Professor Kamal Youcef-Toumi (MIT MECHE)

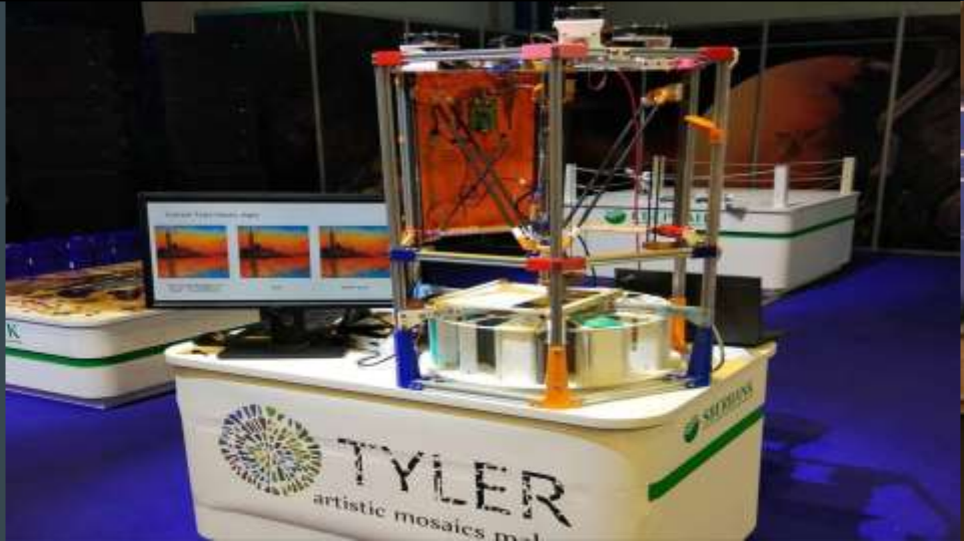
Ph.D. student Dmitry Mironov

(Skoltech ISR)

My Accomplishments

- Cofounder of Tyler - mosaics service (preparing patent)
- Startup fair 1st place - Feb 2018, Skoltech
- Mosgortech contest finalist - Nov 2017, Moscow
- Startup Sauna Russia finalist - Apr 2017, Moscow
- Factory Hack 1st place - Mar 2017, Germany
- “Eureka concept” finalist - Dec 2016, Moscow
- Junction “Intelligent buildings” track & “Smart lighting” challenge winner - Nov 2016, Finland


JUNCTION


TYLER
artistic mosaics


1 m²
≥ 1 week
≥ 500 USD

Thank you!

Middleware position

1. http://moveit.ros.org/documentation/concepts/
2. *http://wiki.ros.org/urdf/Tutorials/Create%20your%20own%20urdf%20file*
3. *P.J. Zsombor-Murray, "Descriptive Geometric Kinematic Analysis of Clavel's "Delta" Robot", 2004*
4. *Matlab simulation toolbox for serial, parallel and hybrid robots 10.1109/ICAR.2017.8023639*
5. *P. J. Zsombor-Murray, Descriptive geometric kinematic analysis of Clavel's "Delta" Robot., Montreal: McGill University, 2004.*
6. *Modeling and control of a Delta-3 robot, Andre Olson, 2009*
7. *S. Kucuk and Z. Bingul, "An off-line robot simulation toolbox," Computer*

Serial vs parallel robots comparison

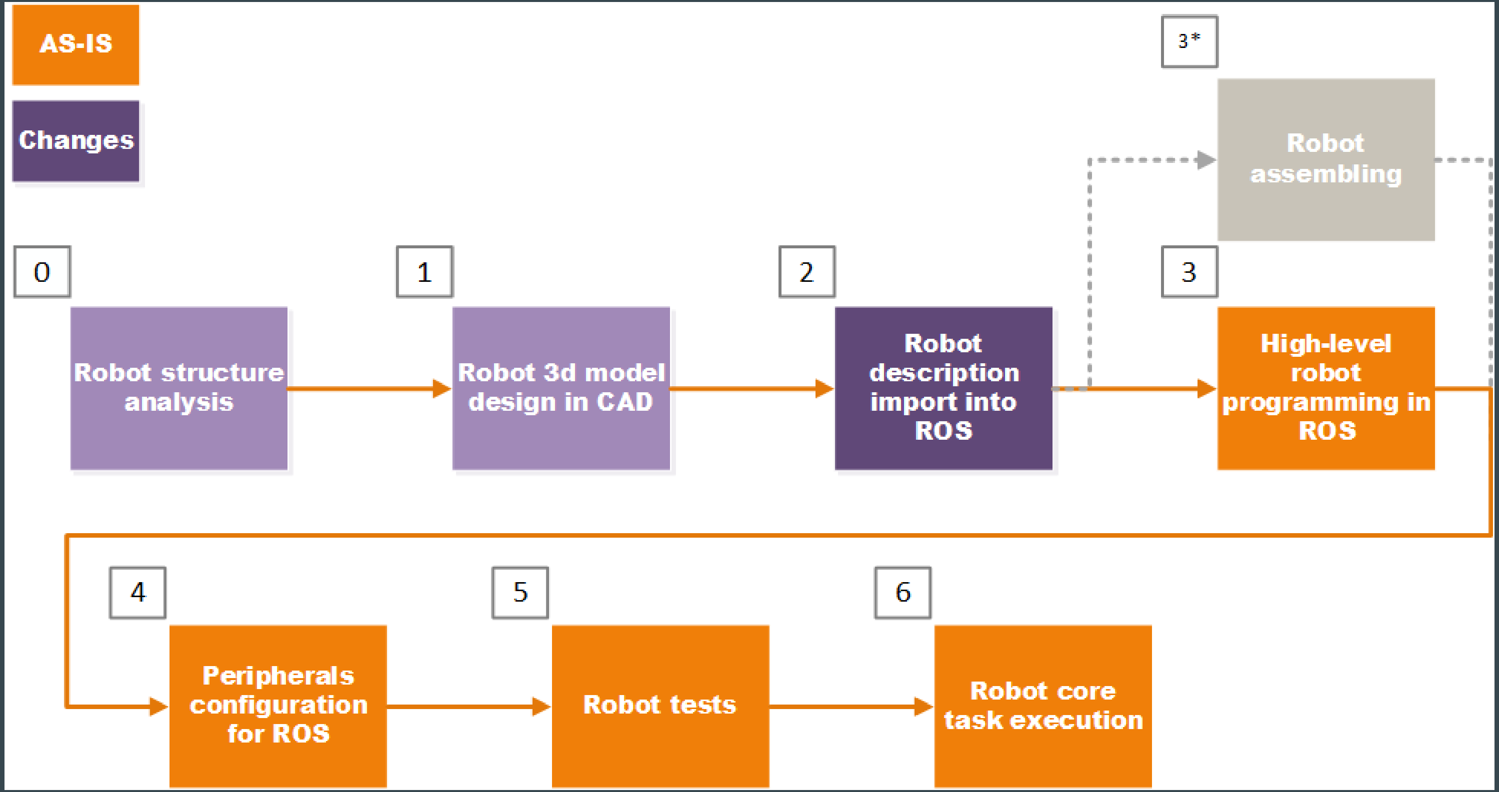
	Parallel robots	Serial robots
Type of kinematic chain	Closed-loop	Open-loop
Workspace	Small and complicated	Large
FK solution	Problematic	Simple
IK solution	Simple	Problematic
Position error	Averages	Accumulates
Force error	Accumulates	Averages
Maximum force	Sum of all motors forces	Limited by minimum actuator force
Stiffness	High	Low
Inertia	Small	Large
Payload/weight ratio	Large	Small
Accuracy	High	Low
Calibration	Complicated	Simple
Workspace/robot size ratio	Small	Large
Motors position	Near fixed base	On links

Middleware position

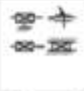
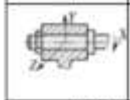
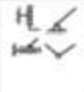
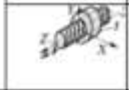
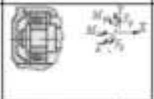
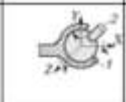
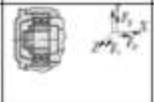
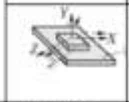
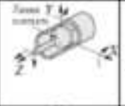
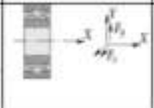
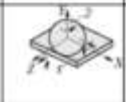
Middleware comparison

Criteria	OpenRTM-aist	OPRos	ROS	Orocos
System model	Component-based framework, model-driven architecture, Platform independent model, platform-specific model	Component-based frameworks, validation/test tools	Component-based framework, publisher/subscribe	C++ libraries: OCL, KDL, BFL
Control model	Component-based	Client/server mechanism for control flow and publisher/subscriber mechanism for data/event flow	Message oriented frameworks	Event-based control
Simulator	OpenHRP3	Yes	Yes	No, Orocos Simulink Toolbox
Behavior coordination	No	No	Yes	No
Distributed environment	Yes	Yes	Yes	No
Robot software architecture indepependent	Yes	Yes	Yes	Yes
Real-time	Yes	No	Planned	Yes
Open-source	Yes	Yes	Yes	Yes
Windows	yes	Partial functions	Yes	Yes
Linux	Yes	Yes	Yes	Yes

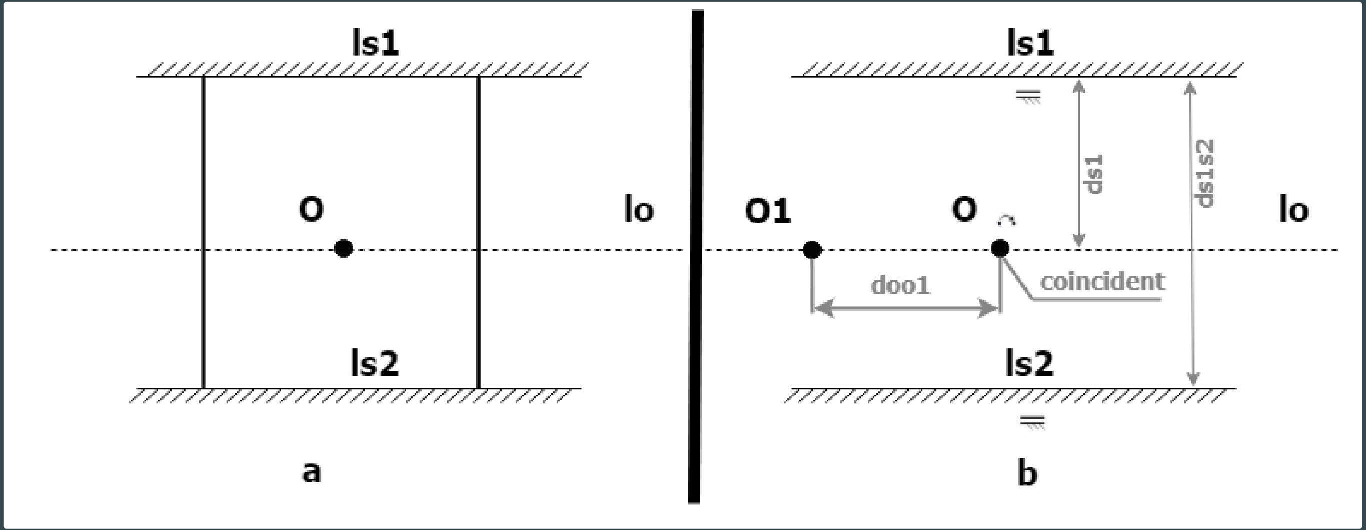
Robot creation workflow



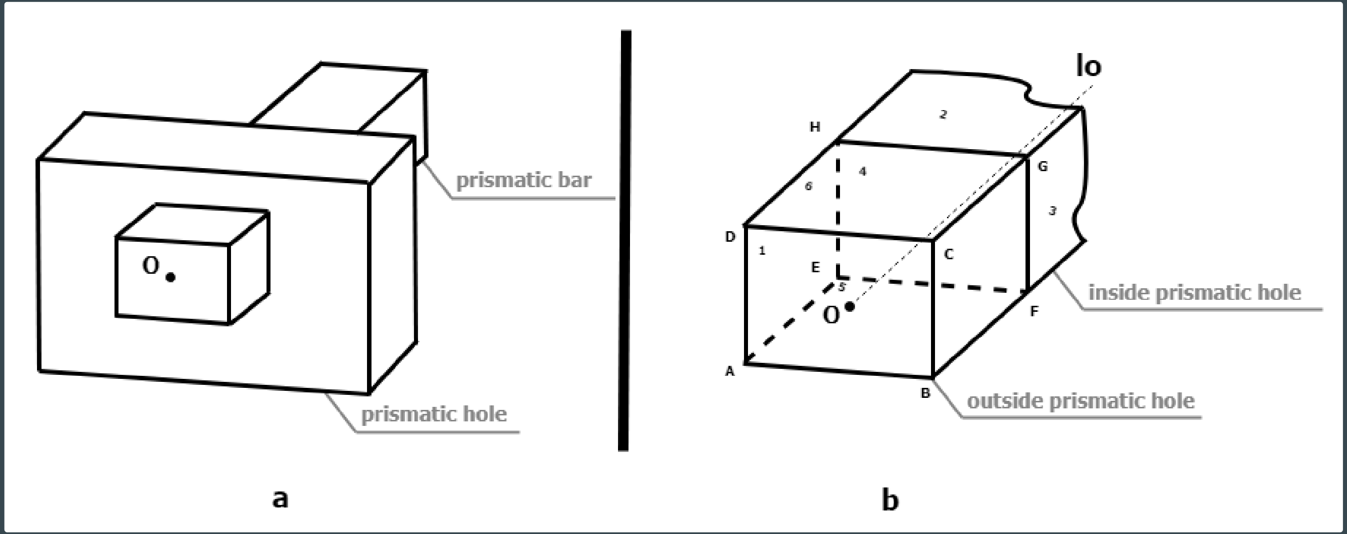
Kinematic pairs classification

Type	Mobility	Link number	3D image	2D image	Joint example and load forces	Joint example name
Lower	I	5				Ball-point slides
						One-tier alignment bearing
						Ball screw
	II	4				Alignment bearing fixed in X axis
	III	3				Spherical bearing fixed in X axis
						Step bearing
Higher	IV	2				Spherical bearing free in X axis
	V	1				Complex

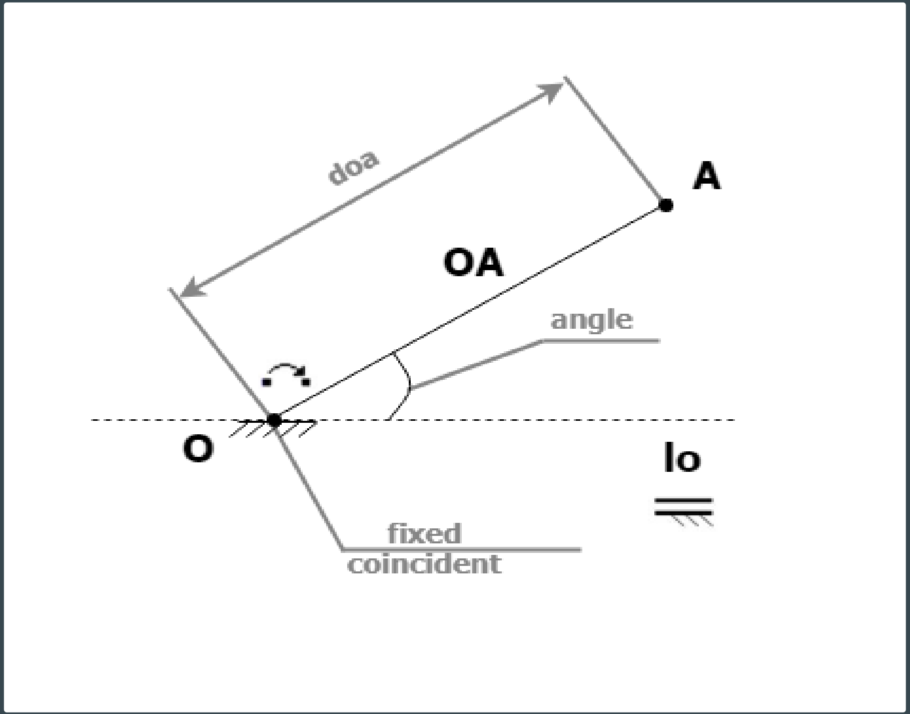
Kinematic pairs constraints analysis



Kinematic pairs constraints analysis



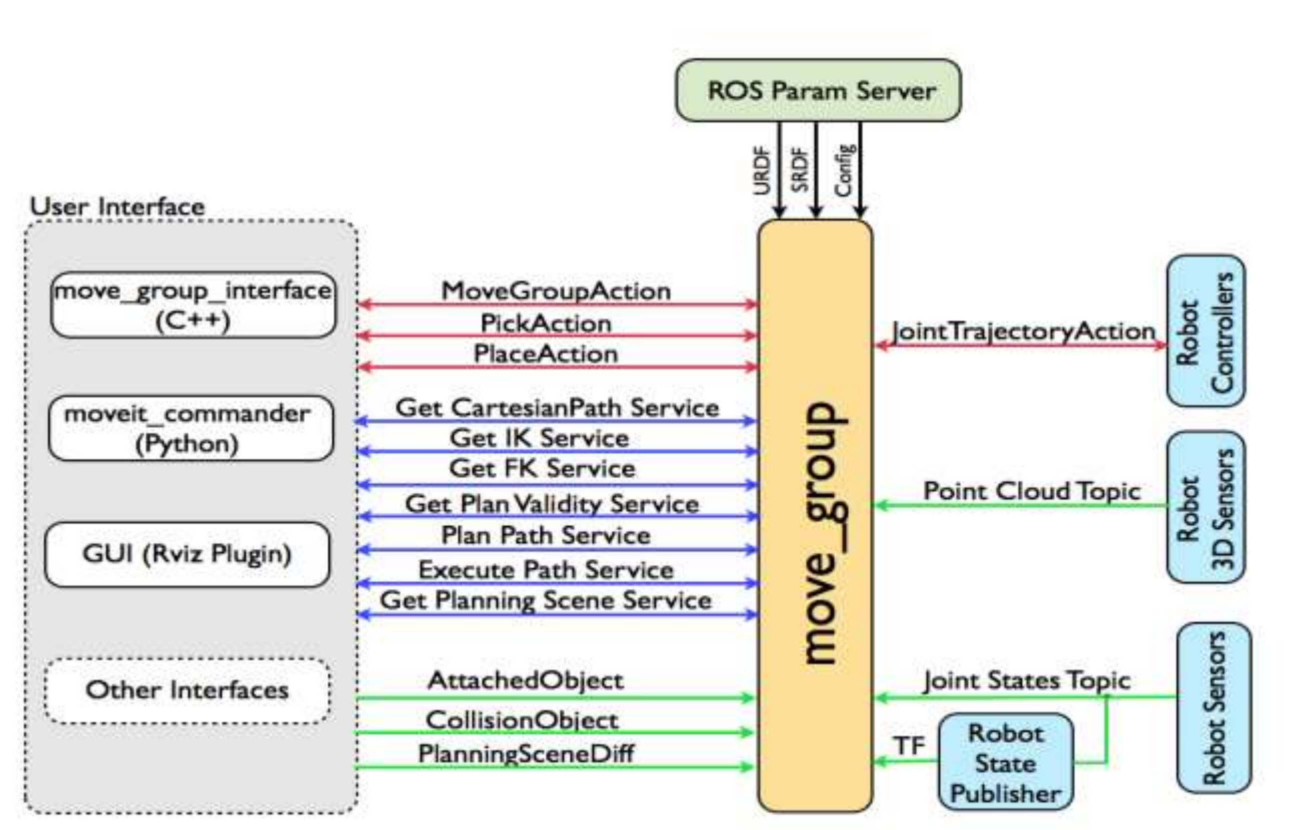
Kinematic pairs constraints analysis



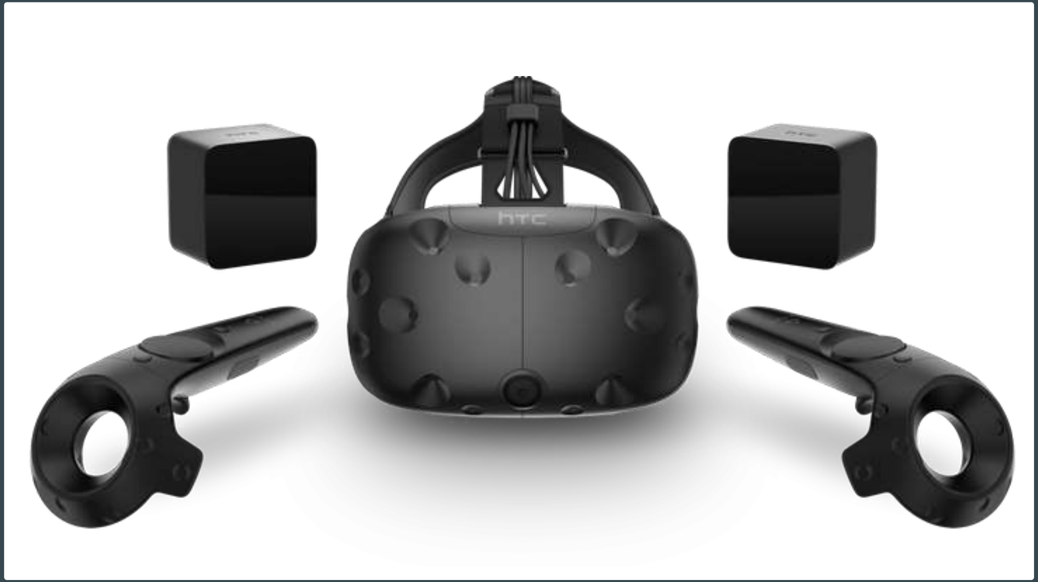
Delta robot GCRDF file

```
<robot name="test_robot">
  <geometry type="fixed", in3d="yes">
    <point3d name="O", xyz="0,0,0"/>
    <point3d name="p1", xyz="0,0,0"/>
    <point3d name="p2", xyz="0,0,0"/>
    <point3d name="p3", xyz="0,0,0"/>
    <point3d name="pF1", xyz="0,0,0"/>
    <point3d name="pF2", xyz="0,0,0"/>
    <point3d name="pF3", xyz="0,0,0"/>

    <line3d name="p1p2", p1="p1", p2="p2"/>
    <line3d name="p1p3", p1="p1", p2="p3"/>
    <line3d name="p2p3", p1="p3", p2="p2"/>

    <line3d name="sl1", p1="p1", p2="pF3"/>
    <line3d name="sl2", p1="p2", p2="pF1"/>
    <line3d name="sl3", p1="p3", p2="pF2"/>
  </geometry>

  <geometry type="moving", in3d="yes">
    <point3d name="pJ1", xyz="0,0,0"/>
    <point3d name="pJ2", xyz="0,0,0"/>
    <point3d name="pJ3", xyz="0,0,0"/>
    <point3d name="pF11", xyz="0,0,0"/>
    <point3d name="pF22", xyz="0,0,0"/>
    <point3d name="pF33", xyz="0,0,0"/>

    <point3d name="pE", xyz="0,0,0"/>
    <point3d name="p11", xyz="0,0,0"/>
    <point3d name="p22", xyz="0,0,0"/>
    <point3d name="p33", xyz="0,0,0"/>

    <line3d name="p11p22", p1="p11", p2="p22"/>
    <line3d name="p22p33", p1="p22", p2="p33"/>
    <line3d name="p11p33", p1="p11", p2="p33"/>
    <line3d name="pF1pJ1", p1="pF1", p2="pJ1"/>
    <line3d name="pF2pJ2", p1="pF2", p2="pJ2"/>
    <line3d name="pF3pJ3", p1="pF3", p2="pJ3"/>
    <line3d name="pF11pJ1", p1="pF11", p2="pJ1"/>
    <line3d name="pF22pJ2", p1="pF22", p2="pJ2"/>
    <line3d name="pF33pJ3", p1="pF333", p2="pJ3"/>
  </geometry>

  <constraints in3d="yes">
    <midpoint3d point="pF11", line="p11p33"/>
    <midpoint3d point="pF22", line="p11p22"/>
    <midpoint3d point="pF33", line="p22p33"/>
    <parallel3d ob1="p11p22", obj2="p1p2"/>
    <parallel3d ob1="p22p33", obj2="p2p3"/>
    <parallel3d ob1="p11p33", obj2="p1p3"/>

    <angle3d name="alfa1" obj1="sl1", obj2="pF3pJ3", value="0.0", mutable="yes"/>
    <angle3d name="alfa2" obj1="sl2", obj2="pF1pJ1", value="0.0", mutable="yes"/>
    <angle3d name="alfa3" obj1="sl3", obj2="pF2pJ2", value="0.0", mutable="yes"/>
  </constraints>
</robot>
```

MoveIt! architecture



HTC Vive Tracking equipment



Experiment setup photo