# ROS Stage simulátor pro Dacep

Ke spuštění simulace navigace pro robota Dacep, je nutné extrahovat složku souborů catkin_navidacep.tar.gz. Ta obsahuje catkin složku s kompletní navigací. Pak stačí provést úpravy popsané v bodu 1. a poté spustit launchfile:

```
$ roscore
--new Terminal--
$ cd .../catkin_navidacep/src/dacep/launch
$ roslaunch dacep-amcl.launch
```

## 1. Navigace: dacep-amcl.launch

Tento skript je tzv. „lunchfile“, to znamená, že startuje jednotlivé „nody“, které sem napíšeme. To zlepšuje přehlednost skriptu a rychlost. Jinak bychom museli každý node startovat příkazem „rosrun“, což je zdlouhavé. Start nodu má v launchfilu tuto syntaxi:

```
<node name="node_name" pkg="package" type="script_name" args="argument, e.g.
path">
```

- node_name se uvádí v každém skriptu nodu u příkazu rospy.init_node('node_name', anonymous=True)
- package je jméno balíčku, jehož je skript součástí
- script_name je název skriptu – např.: script.py, script.cpp,...
- do argumentu podáváme nepříklad cestu k skriptu, který obsahuje daný node

dacep-amcl.launch startuje tyto nody:

- **Stage** – simulátor pod ROS, který je již poměrně zastaralý. Instaluje se při Desktop-Full Install spolu s ROSem. Příjmá data z topicu /cmd_vel a publikuje data ze sonarů a laserů do topiců /base_scan_0 až 7
  - tag <remap> přejmenuje topic. V našem případě /base_laser0 na /scan. V topicu /scan totiž očakává node /move_base data z laserového scanneru.
- **sonar2lasescan** – node, který příjmá topicy /base_scan_1 až 7 (sonary) a směruje je do /SRF08_1 až 7. Zároveň data upravuje do datového typu LaserScan. Topicy /SRF08_1 až 7 jsou použitelné pro navigaci (move_base)
- **joint_state_publisher** – publikuje JointState messages pro všechny nezafixované klouby
- **robot_state_publisher** – používá .urdf soubor, který je definován parametrem v robot_description a kloubové souřadnice z nodu joint_state_publisher, aby vypočetl dopřednou kinematiku (přímá úloha)
- **map_view** – součástí map_server
- **map_publisher** - součástí map_server
- **amcl** – obsahuje pravděpodobnostní lokalizační systém pro robota ve 2D
- **move_base** – hlavní node navigace. Přijímá cíl cesty, tzv. „goal“. Spojuje lokální a globální plánovač tak, aby dosáhl goal. Je kompatibilní s jakýmkoliv globálním plánovačem, který navazuje na nav_core::BaseGlobalPlanner (interface specifikovaný v nav_core) a jakýmkoliv lokálním plánovačem navazujícím na nav_core::BaseLocalPlanner (interface specifikovaný v nav_core). move_base také používá global a local costmap.
  - V tomto nodu lze změnit globální planner, v následujícím nefunguje VoronoiPlanner:

```
<!-- GLOBAL PLANNER: global_planner/GlobalPlanner, carrot_planner/CarrotPlanner,
voronoi_planner/VoronoiPlanner, ... defaultne pouzije posledni pouzity  -->
<param name="base_global_planner" value="global_planner/GlobalPlanner"/>
```

- **recovery** – řekne robotu, ať couvá, pokud dlouho nikam nepopojel a přitom stále nedosáhl cíle
- **rviz** – spouští vizualizační nástroj pro simulaci robota v prostředí. RVIZ se při Desktop-Full Install nainstaluje spolu s ROSem.

V skriptu dacep-amcl.launch je nutné změnit několik argumentů tak, aby byla simulace navigace funkční. Změňme tedy cesty k požadovaným souborům u těchto skriptů:

```
<!-- Read robot model -->
<arg name="model"
default="/home/korys/catkin_navidacep/src/model/urdf/dacep.urdf" />

<!-- Stage -->
<node name="Stage" pkg="stage_ros" type="stageros"
args="/home/korys/catkin_navidacep/src/stage/simple.world">

<!-- Map server -->
<node name="map_view" pkg="map_server" type="map_server"
args="/home/korys/catkin_navidacep/maps/map_actual.yaml" >
```

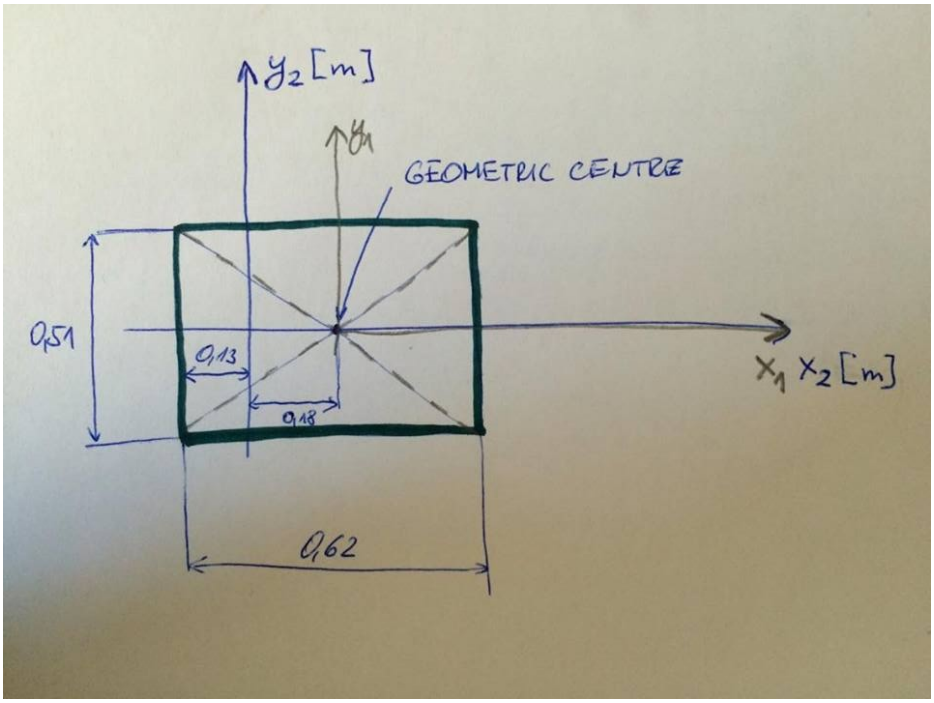
## 2. Soubory pro Stage simulátor: .world a .inc

O Stage simulátoru se lze více dočíst v přiloženém „playerstage_instructions_STAGE320.pdf“. Soubor simple.world je argumentem pro node Stage v našem launchfilu. V simple.world jsou popsány základní vlastnosti simulátoru(prostředí, roboti a další objekty):

- **window** - vlastnosti okna simulátoru
- **floorplan** – vlastnosti mapy simulátoru a cesta k mapě (formát .pgm)
- **dacep** – popis a umístění robota, na začátku musí být „import dacep.inc“, aby tento příkaz fungoval

Soubory .inc obsahují kusy kódu, které lze importovat do worldfile. V „dacep.inc“ jsou příkazy:

- **ranger** – v rámci tohoto příkazu lze definovat jakékoliv zařízení určené k detekci překážek (typ, tvar, pozici)
  - sensor – definuje dosah, rozsah, vzorkování,...
- **block** – definuje tvar pomocí bodů půdorysu (musí následovat za sebou proti směru hodinových ručiček) a výšky objektu
- **size** – velikost „uzavřená do kvádru“
- **color** – barva
- **pose** – umístění (tady pozor na vztažné soustavy)
- **origin** – definuje vzdálednost geometrického středu půdorysu robota a středu otáčení. Od počátku souřadného systému „2“ se definují pozice sensorů. Pokud u „z-souřadnice“ uvedeme 0, pak je počátek umístěn na horní stěně kvádru, který je definován příkazem „size“




Stage simulátor publikuje informace ze senzorů do topiců „base_scan_X“, kde X=0,1,2,... Velikost čísla X záleží na pořadí, v jakém byl uveden senzor ve worldfile nebo incfile. V našem případě je nejdříve uveden laserový scanner hokuyo a až po něm jednotlivé sonary. To znamená, že laser bude publikován do topicu „base_scan_0“ a senzory do „base_scan_1 až 7“. Zprávy se publikují ve formátu LaserScan, nicméně sonary mají nadefinován pouze 1 vzorek pro celý 15° rozsah. Proto je nutné data pro zobrazení v RVIZu upravit skriptem sonar2lasescan.py.

## 3. Soubory pro mapy: .yaml, .pgm, .png

Mapa je zadána grafickým souborem .pgm nebo .png. Vlastnosti mapy by měly být nastaveny v souboru .yaml, který obsahuje tyto specifikace:

- **image –** cesta k souboru .pgm, .png obsahujícím mapu
- **resolution** – kolik metrů měří jeden pixel v mapě
- **origin** – 2D poloha levého dolního pixelu v mapě [x, y, yaw]. Yaw je rotace proti směru hodonových ručiček.
- **negate** – invertování barev černá/bílá a tím změna významu překážky/volného místa
- **free_tresh** – horní práh odstínu šedi (pravděpodobnosti výskytu překážky), který je interpretován jako volné místo
- **occupied_tresh** - dolní práh odstínu šedi (pravděpodobnosti výskytu překážky), který je interpretován jako překážka

## 4. Soubor modelu .urdf

Lze z něj vyčíst celou geometrii modelu, rozměry a umístění senzorů.