# Параметры командной строки

**-port0:** указывается номер сетевого порта для управления, по умолчанию 8888

**-logfile:** указывается путь и имя файла для записи лога, по умолчанию - *simulation3dlog.txt*

Пример:

**simulation3d.exe -port:7777 logfile:d:/folder/log.txt**

# Управление программой

F10 – выход

Управление камерой: сдвиги - движение мышью удерживая левую кнопку, повороты - движение мышью удерживая правую кнопку, наезд-отъезд камеры – прокручиванием колёсика мыши.

# Сетевой протокол

Cлушается порт управления.

**Диаграммы последовательности протокола управления.**

соединение

ready

Клиент

Сервер

**. . .**

команда

ready + результат

**. . .**

рассоединение

end

Пакеты передаются в формате **json**. Перед каждым пакетом находится заголовочная строка, например: 'json123:', где цифры '123' - размер в байтах пакета следующего после символа ':'

В скрипте *client0.py* – примеры управление через сокеты.

**Формат пакетов управления.**

*ready*

{

"packet":"ready"

}

У пакета ready возможны дополнительные поля с результатами выполнения команд.

Команда *create manipulator1*

Масса подразумевается в килограммах.

Размеры подразумеваются в метрах, углы в градусах.

Смысл размеров - в чертежике для манипулятора1.

Для звеньев, которые управляются моторами задаются коэф. САР.

САР – система автоматического регулирования

Коэффициенты: ACSProportional, ACSIntegral, ACSDifferential

В режиме кинематики позиционир. не САР, а KinematicAngularVelocity

{

"packet":"create",

"type":"manipulator1",

"x":<float>, *(необязательный, по умолчанию 0.0)*

"y":< float >, *(необязательный, по умолчанию 0.0)*

"z":< float >, *(необязательный, по умолчанию 0.0)*

"angle":< float >, *(необязательный, по умолчанию 0.0)*

"Kinematic":< bool >, *(необязательный, по умолчанию false)*

"KinematicAngularVelocity":< float >, *(необязательный, по умолчанию 100.0)*

"UseGravity":< bool >, *(необязательный, по умолчанию false)*

"ChassisHeight":< float >, *(необязательный, по умолчанию 1.0)*

"ChassisWidth":< float >, *(необязательный, по умолчанию 0.5)*

"RotatingplatformMass":< float >, *(необязательный, по умолчанию 1.0)*

"RotatingplatformACSProportional":< float >, *(необяз., по умолчанию 1.5)*

"RotatingplatformACSIntegral":< float >, *(необяз., по умолчанию 0.0)*

"RotatingplatformACSDifferential":< float >, *(необяз., по умолчанию 1.1)*

"RotatingplatformAngle0":< float >, *(необязательный, по умолчанию 0.0)*

"RotatingplatformAngle1":< float >, *(необязательный, по умолчанию 360.0)*

"RotatingplatformDiameter":< float >, *(необязательный, по умолчанию 0.5)*

"RotatingplatformWidth":< float >, *(необязательный, по умолчанию 0.02)*

"HolderMass":< float >, *(необязательный, по умолчанию 1.0)*

"HolderWidth":< float >, *(необязательный, по умолчанию 0.03)*

"HolderLength":< float >, *(необязательный, по умолчанию 0.14)*

"LeverMass":< float >, *(необязательный, по умолчанию 1.0)*

"LeverACSProportional":< float >, *(необязательный, по умолчанию 1.5)*

"LeverACSIntegral":< float >, *(необязательный, по умолчанию 0.0)*

"LeverACSDifferential":< float >, *(необязательный, по умолчанию 1.1)*

"LeverAngle0":< float >, *(необязательный, по умолчанию 10.0)*

"LeverAngle1":< float >, *(необязательный, по умолчанию 60.0)*

"LeverhingeMass":< float >, *(необязательный, по умолчанию 1.0)*

"LeverhingeDiameter":< float >, *(необязательный, по умолчанию 0.08)*

"LeverhingeWidth":< float >, *(необязательный, по умолчанию 0.023)*

"LeverWidth":< float >, *(необязательный, по умолчанию 0.03)*

"LeverLength":< float >, *(необязательный, по умолчанию 0.6)*

"ArmMass":< float >, *(необязательный, по умолчанию 1.0)*

"ArmACSProportional":< float >, *(необязательный, по умолчанию 1.5)*

"ArmACSIntegral":< float >, *(необязательный, по умолчанию 0.0)*

"ArmACSDifferential":< float >, *(необязательный, по умолчанию 1.1)*

"ArmAngle0":< float >, *(необязательный, по умолчанию 60.0)*

"ArmAngle1":< float >, *(необязательный, по умолчанию 120.0)*

"ArmhingeMass":< float >, *(необязательный, по умолчанию 1.0)*

"ArmhingeDiameter":< float >, *(необязательный, по умолчанию 0.08)*

"ArmhingeWidth":< float >, *(необязательный, по умолчанию 0.023)*

"ArmWidth":< float > *(необязательный, по умолчанию 0.03)*

"ArmLength":< float > *(необязательный, по умолчанию 0.6)*

}

Ответ для команды *create manipulator1* : пакет *ready* с параметром.

{

"packet":"ready",

"id":<целое> (идентификатор устройства, 0 – если ошибка)

}

Команда *create manipulator2*

Масса подразумевается в килограммах.

Размеры подразумеваются в метрах, углы в градусах.

Смысл размеров - в чертежиках для манипулятора2, **захвата и пальца**.

Для звеньев, которые управляются моторами задаются коэф. САР.

САР – система автоматического регулирования

Коэффициенты: ACSProportional, ACSIntegral, ACSDifferential

В режиме кинематики позиционир. не САР, а KinematicAngularVelocity

{

"packet":"create",

"type":"manipulator1",

"x":<float>, *(необязательный, по умолчанию 0.0)*

"y":< float >, *(необязательный, по умолчанию 0.0)*

"z":< float >, *(необязательный, по умолчанию 0.0)*

"angle":< float >, *(необязательный, по умолчанию 0.0)*

"Kinematic":< bool >, *(необязательный, по умолчанию false)*

"KinematicAngularVelocity":< float >, *(необязательный, по умолчанию 100.0)*

"UseGravity":< bool >, *(необязательный, по умолчанию false)*

"ChassisHeight":< float >, *(необязательный, по умолчанию 1.0)*

"ChassisWidth":< float >, *(необязательный, по умолчанию 0.5)*

"RotatingplatformMass":< float >, *(необязательный, по умолчанию 1.0)*

"RotatingplatformACSProportional":< float >, *(необяз., по умолчанию 1.5)*

"RotatingplatformACSIntegral":< float >, *(необяз., по умолчанию 0.0)*

"RotatingplatformACSDifferential":< float >, *(необяз., по умолчанию 1.1)*

"RotatingplatformAngle0":< float >, *(необязательный, по умолчанию 0.0)*

"RotatingplatformAngle1":< float >, *(необязательный, по умолчанию 360.0)*

"RotatingplatformDiameter":< float >, *(необязательный, по умолчанию 0.5)*

"RotatingplatformWidth":< float >, *(необязательный, по умолчанию 0.02)*

"LeverhingeMass":< float >, *(необязательный, по умолчанию 1.0)*

"LeverhingeDiameter":< float >, *(необязательный, по умолчанию 0.08)*

"LeverMass":< float >, *(необязательный, по умолчанию 1.0)*

"LeverACSProportional":< float >, *(необязательный, по умолчанию 1.5)*

"LeverACSIntegral":< float >, *(необязательный, по умолчанию 0.0)*

"LeverACSDifferential":< float >, *(необязательный, по умолчанию 1.1)*

"LeverWidth":< float >, *(необязательный, по умолчанию 0.03)*

"LeverLength":< float >, *(необязательный, по умолчанию 0.6)*

"ArmMass":< float >, *(необязательный, по умолчанию 1.0)*

"ArmACSProportional":< float >, *(необязательный, по умолчанию 1.5)*

"ArmACSIntegral":< float >, *(необязательный, по умолчанию 0.0)*

"ArmACSDifferential":< float >, *(необязательный, по умолчанию 1.1)*

"ArmAngle0":< float >, *(необязательный, по умолчанию 60.0)*

"ArmAngle1":< float >, *(необязательный, по умолчанию 120.0)*

"ArmhingeMass":< float >, *(необязательный, по умолчанию 1.0)*

"ArmhingeDiameter":< float >, *(необязательный, по умолчанию 0.08)*

"ArmhingeWidth":< float >, *(необязательный, по умолчанию 0.023)*

"ArmWidth":< float >, *(необязательный, по умолчанию 0.03)*

"ArmLength":< float >, *(необязательный, по умолчанию 0.9)*

"HolderMass":< float >, *(необязательный, по умолчанию 1.0)*

"HolderWidth":< float >, *(необязательный, по умолчанию 0.03)*

"HolderLength":< float >, *(необязательный, по умолчанию 0.14)*

"HoldersDistance":< float >, *(необязательный, по умолчанию 0.179)*

"WheelhingeMass":< float >, *(необязательный, по умолчанию 1.0)*

"WheelhingeWidth":< float >, *(необязательный, по умолчанию 0.046)*

"WheelMass":< float >, *(необязательный, по умолчанию 1.0)*

"WheelDiameter":< float >, *(необязательный, по умолчанию 0.32)*

"WheelWidth":< float >, *(необязательный, по умолчанию 0.023)*

"WheelLever":< float >, *(необязательный, по умолчанию 0.12)*

"LeverAngle0":< float >, *(необязательный, по умолчанию 10.0)*

"LeverAngle1":< float >, *(необязательный, по умолчанию 60.0)*

**(Todo: заполнить поля)**

**…**

}

Ответ для команды *create manipulator2* : пакет *ready* с параметром.

{

"packet":"ready",

"id":<целое> (идентификатор устройства, 0 – если ошибка)

}

Команда *delete*, удаляет устройство*.*

{

"packet":"delete",

"id":<целое> (идентификатор устройства)

}

Ответ для команды *delete* : пакет *ready* с параметром.

{

"packet":"ready",

"ok":<целое> (1 - успешно, 0 – если ошибка)

}

Команда *clear*, удаляет все устройства*.*

{

"packet":"clear",

}

Ответ для команды *clear* : пакет *ready* с параметром.

{

"packet":"ready",

"ok":<целое> (1 - успешно, 0 – если ошибка)

}

Команда *setpos*, устанавливает позиции плечей манипуляторов типа *manipulator1* или *manipulator2.* Углы – в градусах. Смысл - в чертежиках для манипуляторов 1 и 2.

{

"packet":"setpos",

"id":<целое> (идентификатор устройства)

"a0":<целое> (угол поворота платформы)

"a1":<целое> (угол поворота нижней секции)

"a2":<целое> (угол поворота верхней секции)

}

***Для команды setpos ответ не приходит!***

Команда *setgripper*, выполняет захват или отпускает захват.

{

"packet":"setgripper",

"id":<целое> (идентификатор устройства)

"gripped":<целое> (1 – выполнить захват, 0 – отпустить захват)

}

Ответ для команды *setgripper*: пакет *ready* с параметром.

{

"packet":"ready",

"ok":<целое> (1 – есть захват, 0 – нет захвата, -1 – если ошибка)

}

Команда *setcamera*, устанавливает позицию камеры.

Размеры подразумеваются в метрах.

{

"packet":"setcamera",

"x0":<целое> (координата x точки, на которую направлена камера)

"y0":<целое> (координата y точки, на которую направлена камера)

"z0":<целое> (координата z точки, на которую направлена камера)

"x1":<целое> (координата x камеры)

"y1":<целое> (координата y камеры)

"z1":<целое> (координата z камеры)

}

Ответ для команды *setcamera*: пакет *ready* с параметром.

{

"packet":"ready",

"ok":<целое> (1 - успешно, 0 – если ошибка)

}