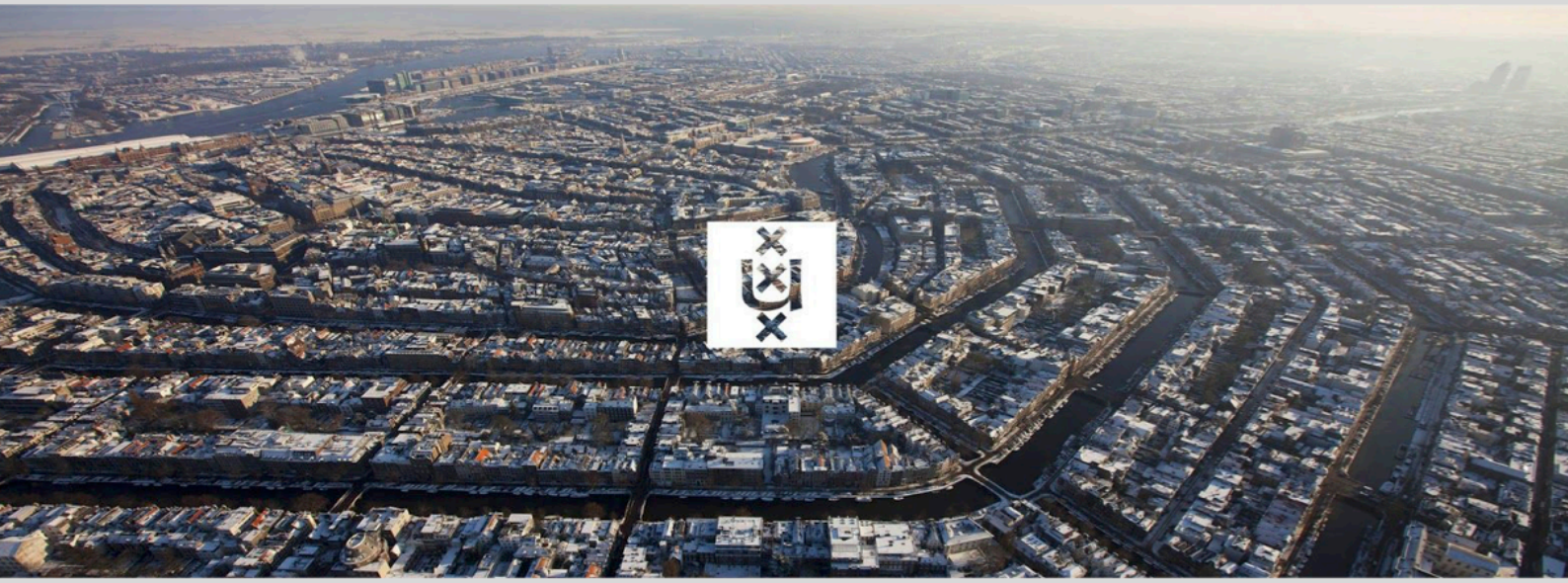


Institute for Logic, Language and Computation

A Short Enki Tutorial

Shuai Wang

ILLC, UvA

Contents

ABSTRACT

This tutorial is for those who need a quick guidance of the Enki robot simulation platform. Most of the examples included are from the online Enki reference document and the examples included [1].

[1] http://lis2.epfl.ch/resources/download/doc1.0/libenki/index.html

1 INTRODUCTION

Enki[2] is a robot simulator. All robots are taken as objects and are placed in a virtual world. Some are static objects and some others are agents/robots and may interact with the environment. In the bare Enki distribution, there are four kinds of robots included (three true robots and beacon).

1.1 ROBOTS

1.2 EPUCK

The e-puck robot[3] has been developed as an open tool.

1.3 KHEPERA

The Khepera is a small (5.5 cm) mobile robot that is out of date.

1.4 SBOT

Swarm robots also developed by EPFL.

1.5 SBOTACTIVEOBJECT

A SbotActiveObject object is a beacon for the S-bot experiments **

2 A FIRST EXAMPLE

2.1 A ROBOT AND A WORLD

We create a world and initialise it with a robot.

```
#include <enki/PhysicalEngine.h>
#include <enki/robots/e-puck/EPuck.h>
#include <iostream>

int main(int argc, char *argv[])
{
        // Create the world
```

[2] https://github.com/enki-community/enki
[3] http://www.e-puck.org/

```
        Enki::World world(200, 200);

        // Create a robot and position it
        Enki::EPuck *ePuck = new Enki::EPuck;
        ePuck->pos = Enki::Point(100, 100);
        ePuck->leftSpeed = 30;
        ePuck->rightSpeed = 20;

        // objects are garbage collected by the world on destruction
        world.addObject(ePuck);

        // Run for some times
        for (unsigned i=0; i<10; i++)
        {
                // step of 50 ms
                world.step(0.05);
                std::cout << "E-puck pos is (" << ePuck->pos.x
                << "," << ePuck->pos.y << ")" << std::endl;
        }
}
```

2.2 OBJECT

The following is a world with an polygone object in the middle of the world.

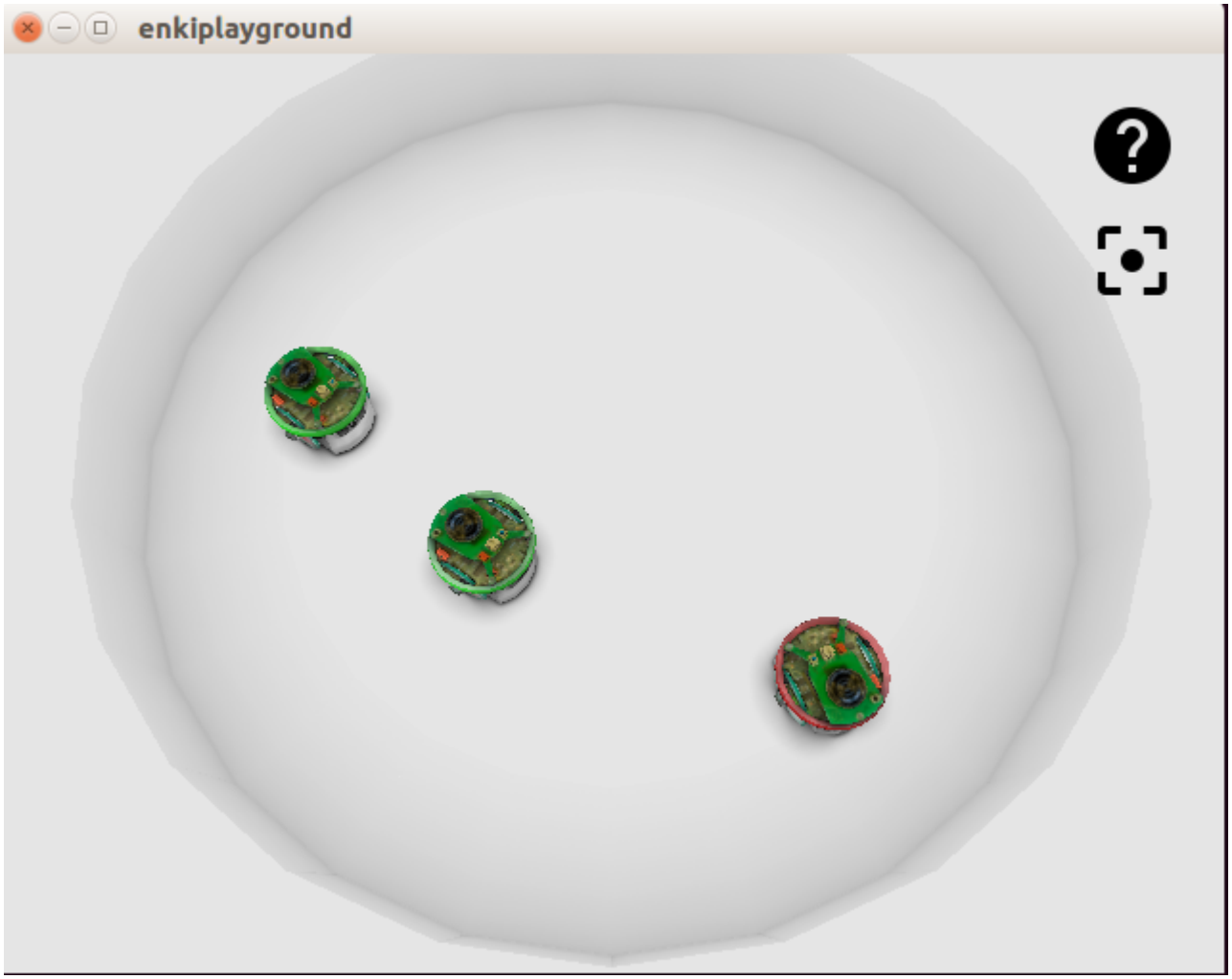
```
#include <enki/PhysicalEngine.h>
#include <enki/robots/e-puck/EPuck.h>
#include <iostream>
#include <viewer/Viewer.h>

int main(int argc, char *argv[])
{
        // Create the world
        Enki::World world(200, 200);

        // Create a Khepera and position it
        Enki::EPuck *ePuck = new Enki::EPuck;
        ePuck->pos = Enki::Point(100, 100);
        ePuck->leftSpeed = 30;
```

```
        ePuck->rightSpeed = 20;

        // objects are garbage collected by the world on destruction
        world.addObject(ePuck);

        Enki::Polygone p;
        const double amount = 9;
        const double radius = 5;
        const double height = 20;
        // std::cout <<"M_PI = " <<M_PI <<std::endl;
        for (double a = 0; a < 2*M_PI; a += 2*M_PI/amount)
                p.push_back(Enki::Point(radius * cos(a),
                radius * sin(a)));
        Enki::PhysicalObject* o = new Enki::PhysicalObject;
        Enki::PhysicalObject::Hull
             hull(Enki::PhysicalObject::Part(p, height));
        o->setCustomHull(hull, 1);
        o->pos = Enki::Point(100, 100);
        o->setColor(Enki::Color(0.4,0.6,0.8));
        world.addObject(o);

        // Run for some times
        for (unsigned i=0; i<10; i++)
        {
                // step of 50 ms
                world.step(0.05);
                std::cout << "E-puck pos is (" << ePuck->pos.x
                << "," << ePuck->pos.y << ")" << std::endl;
        }
}
```

2.3 VIEWER

Enki uses Qt5[4] for simulation interface. We can define a class as an inheritance of the ViewerWidget class.

[4] http://doc.qt.io/qt-5/




```
#include "Viewer.h"
#include <enki/PhysicalEngine.h>
#include <enki/robots/e-puck/EPuck.h>
#include <enki/robots/marxbot/Marxbot.h>
#include <enki/robots/thymio2/Thymio2.h>
#include <QApplication>
#include <QtGui>
#include <iostream>


using namespace Enki;
using namespace std;


class EnkiPlayground : public ViewerWidget
{
protected:
        QVector<EPuck*> epucks;
```

```
        QMap<PhysicalObject*, int> bullets;

public:
        EnkiPlayground(World *world, QWidget *parent = 0) :
                ViewerWidget(world, parent)
        {
                EPuck *epuck = new EPuck;
                epuck->pos = Point(20, 15);
                epuck->leftSpeed = 4;
                epuck->rightSpeed = 5;
                world->addObject(epuck);

                epuck = new EPuck;
                epuck->pos = Point(20, -10);
                epuck->leftSpeed = 5;
                epuck->rightSpeed = 2;
                epuck->setColor(Color(1, 0, 0));
                world->addObject(epuck);

                epuck = new EPuck;
                epuck->pos = Point(0, 30);
                epuck->leftSpeed = 2;
                epuck->rightSpeed = 3;
                epuck->setColor(Color(0, 1, 0));
                world->addObject(epuck);

        }
};

int main(int argc, char *argv[])
{
        QApplication app(argc, argv);

        // Create the world and the viewer
        bool igt(app.arguments().size() > 1);

        QImage gt;

        World world(35, Color(0.9, 0.9, 0.9), World::GroundTexture());
```

```
        EnkiPlayground viewer(&world);

        viewer.show();

        return app.exec();
}
```

ACKNOWLEDGEMENT

I would like to thank Dr. Wei Li his kind help.

References

https://github.com/enki-community/enki