Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра “Системи автоматизованого проектування”

**ЗВІТ**

про Практику за темою бакалаврської кваліфікаційної роботи

студента ІV курсу групи КНз-41

Чалого Михайла Васильовича

**База практики** ТзоВ «ЕЛЕКС»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(назва та адреса підприємства)

**Термін практики** з « » квітня 2016 р. до « » квітня 2016 р.

**Керівники практики**

від кафедри \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_к.т.н., доцент, Фармага І.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (наук.ст, вч.звання, прізвище, ім’я, по батькові)

від бази практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (посада, прізвище, ім’я, по батькові)

**Керівник бакалаврської кваліфікаційної роботи**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_к.т.н., Марікуца У.Б.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (наук.ст, вч.звання, прізвище, ім’я, по батькові)

Тема: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ОЦІНКА** \_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дата** \_\_\_\_\_\_\_\_\_\_\_\_\_

Львів – 2016

# **Зміст**

Зміст 2

Вступ 3

Опис бази практики 5

Коротка характеристика закладу проходження практики 5

Індивідуальне завдання 7

Дослідження об’єкта проектування 7

Актери і юзкейси 8

Natural User Interface (NUI) 10

Технічне рішення 11

Вибір реалізації інтерфейсу користувача 11

Скорочення часу очікування відповіді 11

Підтримка безшовної взаємодії 12

Концепція технічного рішення 12

Концепція фізичного рішення 14

Реалізація 15

Лістінг 1. Program.cs 15

Лістінг2. PositionAndSkeletonEmitter.cs 16

Список використаних джерел 20

# Вступ

Проектно-технологічнапрактика студентів є невід’ємною складовою освітньо-професійної програми підготовки студентів вищих навчальних закладів України. Вона становить важливу та обов’язкову ланку в системі наскрізної практичної підготовки висококваліфікованих фахівців до майбутньої практичної діяльності. Проектно-технологічна практика спрямована на закріплення теоретичних знань, отриманих студентами за час навчання, та вдосконалення практичних навичок і умінь в роботі за обраною спеціальністю.

**Мета** проектно-технологічної практики – поглиблене ознайомлення студентів зі специфікою майбутньої спеціальності, розширення й доповнення знань, здобутих студентами під час вивчення теоретичних курсів та виконання практичних і лабораторних завдань з дисциплін “Алгоритмізація та програмування”, “Проблемно-орієнтоване програмування”, “Основи інформаційних технологій”, “Системне програмування”, “Операційні системи”, “Теорія алгоритмів» Організація баз даних та знань”, “Схемотехніка комп’ютерів”, “Методи та засоби КІТ”, “Архітектура комп’ютерів”, “Комп’ютерні мережі”, “Чисельні методи в інформатиці”, “Моделювання систем”, “Організація баз даних та знань”, “Системи штучного інтелекту” та інших, а також формування професійних умінь і навичок для прийняття самостійних рішень і розв’язування задач пов’язаних із сучасними комп’ютерними інформаційними технологіями в реальних науково-дослідницьких або виробничих умовах, виховання потреби систематично оновлювати свої знання та творчо застосовувати їх у практичній діяльності.

Основними **завданнями** проектно-технологічної практики є:

* ознайомлення з нормативно-технологічною документацією підприємства – бази практики, вивчення особливостей адміністративної та інформаційної взаємодії між його окремими підрозділами;
* ознайомлення з конфігурацією та архітектурою технічних засобів підприємства;
* ознайомлення з програмним забезпеченням і програмними продуктами, які використовуються або створюються в процесі діяльності даного підприємства;
* ознайомлення з технологічними процесами керування інформаційними потоками, організацією зберігання даних, створення баз даних і доступу до даних;
* вивчення послідовності, методів і засобів оброблення даних;
* опанування технологіями розроблення, впровадження та супроводу програмного забезпечення;
* вивчення основ конфігурування технічних засобів, інформаційних систем та мереж, систем автоматизованого проектування та керування;
* оволодіння сучасними технологіями оброблення інформації та застосування їх для виконання практичних завдань;
* набуття виробничих навичок, а саме: технічної, проектувальної, виконавської;
* набуття навичок групової роботи та роботи в колективі спеціалістів;

# Опис бази практики

## Коротка характеристика закладу проходження практики

* Адреса: місто Львів, вул. Наукова, 7г
* Email: office@eleks.com
* Телефон: +38 032 297 12 51
* Факс: +380 32 244-7002

ELEKS - українська компанія, яка займається наданням послуг у сфері розробки та тестування програмного забезпечення. Головний офіс компанії розташований у Львові. Центри розробки - у Івано-Франківську, Тернополі і Кременчуку. Має три закордонні представництва у США: у Лас-Вегасі, Бостоні та південній Флориді.

* Компанія заснована в 1991 році Олексієм Івановичем Скрипником та Олексієм Олексійовичем Скрипником (батько та син). Перші роки свого існування компанія займалася розробкою власних програмних продуктів для електроенергетичних підприємств. З 1997 року компанія почала надавати послуги з розробки програмного забезпечення на замовлення. З 1999 року компанія працює із закордонними замовниками (переважно США, Європа).
* ELEKS Software – одна з найбільших ІТ компаній в Україні з головним офісом у Львові. Сьогодні компанія налічує біля 1000 працівників і надає послуги клієнтам зі всього світу. Компанія має статус Золотого партнера компанії Microsoft, бізнес-партнера IBM та технологічного партнера Borland. Накопичений досвід, талановиті працівники та здатність надавати послуги високої якості дають Елексу можливість робити особливий внесок у бізнес своїх клієнтів і допомагати їм у досягненні їх стратегічних цілей.
* ELEKS Software пропонує повний цикл розробки програмного забезпечення, а також супроводу проектів на всіх стадіях їх життєвого циклу, починаючи від дизайну та розробки і завершуючи тестуванням і підтримкою.
* Послуги компанії включають:
* - Розробку і підтримку програмного забезпечення
* - Послуги із тестування і локалізації програмного забезпечення
* - Створення і підтримку веб-сайтів
* - Послуги в області веб-дизайну та мультимедійних рішень
* - Віддалене адміністрування ІТ-інфраструктури
* - ІТ-аутсорсинг
* - ІТ-консалтинг
* Також компанія продає власні продукти, а саме: Медичну Інформаційну Систему “Доктор Елекс” (http://doctor.eleks.com).

# Індивідуальне завдання

## Дослідження об’єкта проектування

Для підтвердження гіпотези бакалаврської теми було створено систему для керування відображенням інформації портфоліо за допомогою природніх жестів.

Ілюстративний основний плин взаємодії користувача з системою показано на діаграмі нижче.

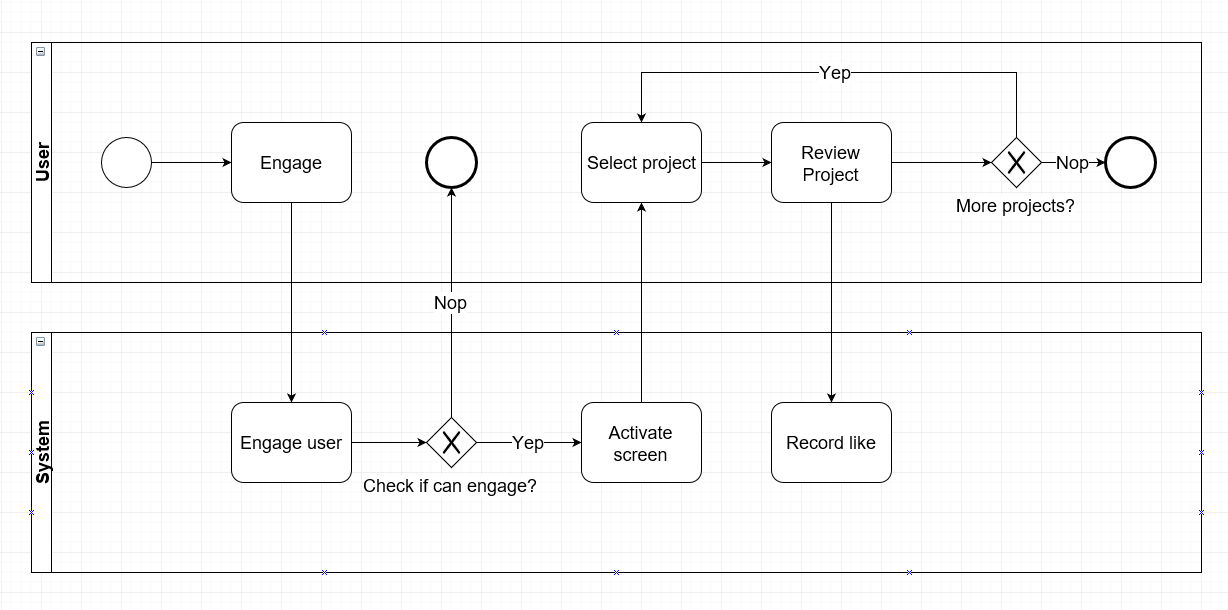


Рис. 1. Плин взаємодії користувача і системи

## Актери і юзкейси

Секція описує основних авторів і пов’язані з ними юзкейси.

Базова реалізація системи розділяє користувачів на три основні актори.

|  |  |
| --- | --- |
| Користувачі | Сторонні люди які користуватимуться системою |
| Адміністратор | Персонал який налаштовує інсталяцію |
| Керівник контенту | Відповідальний за контент |



Рис. 2. Основні юз кейси користувача

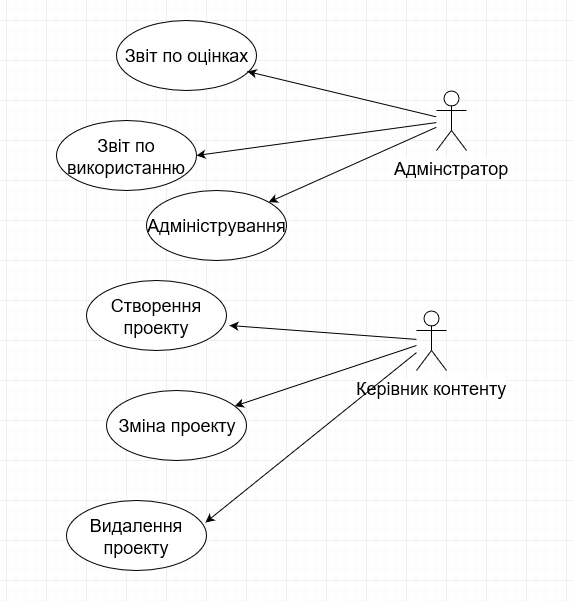


Рис. 3. Основні юз кейси адміністратора і керівника контентом

## Natural User Interface (NUI)

Основна ідея системии це використання NUI для керування інтефресом користувача.

Моделі NUI дозволяють користувачам взаємодіяти з пристроєм за допомогою дій, які є природними, інтуїтивно зрозумілим і загальним для повсякденного людської поведінки. Описане рішення працює за допомогою датчиків Kinect, що можуть реагувати на людські жести і рухи тіла, забезпечуючи негайний і плавний відгук системи. Розпізнавання жестів в рамках NUI взаємодій допомагають впроваджувати нові підходи до інтерактивної роботи користувачів в тих областях, які вимагають залучення аудиторії, такі як освіта, розваги, зв'язків з громадськістю та маркетинг. Презентацій, тренінги, а також рекламні заходи можуть стати дуже інтерактивним за допомогою використання NUI.  
  
З цією метою цей проект використовує технологію розпізнавання руху Microsoft Kinect здля підвищення участі аудиторії в Behance Portfolio Review що, проводиться для Львівської спільноти дизайнерів. Цей захід було ініційовано командою UX Eleks з метою об'єднання творчих професіоналів для обміну знаннями та створення мереж.

## Технічне рішення

### Вибір реалізації інтерфейсу користувача

Щоб створити негайну відповідь для користувачів, ми повинні були визнати, жести якомога швидше і підтримувати зв'язок з інтерфейсом з найнижчою затримкою. Перше завдання процесу розробки призначеного для користувача інтерфейсу було вибрати базову технологію. Найбільш поширеним способом для розробки додатків з багатим інтерфейсом, які можуть взаємодіяти з Kinect є використання Windows Presentation Foundation (WPF) або, графічна підсистема візуалізації призначених для користувача інтерфейсів в додатках на базі Windows за допомогою Microsoft.

WPF це дуже потужна бібліотека UI, що дозволяє створювати практично будь-який парадигми призначеного для користувача інтерфейсу, проте, якщо вам потрібні спеціальні елементи управління, стандартна бібліотека може мати їх. Наприклад, якщо вам потрібно, щоб створити ефект паралакса для фону, немає майже ніяких доступних варіантів, на відміну від HTML, який пропонує безліч добре розроблених і легко підтримуваних бібліотек. Таким чином, ми замінили WPF з HTML5 для призначеного для користувача інтерфейсу, і вирішив використовувати AngularJs як середовище розробки програми.

Скорочення часу очікування відповіді

Інший технічним завданням було встановлення зв'язку між додатком і драйвером додатки HTML. Очевидно, що нам потрібно, щоб це було дуже швидко, так як будь-яка затримка між жестом і рухом фактичного відгуку призначеного для користувача інтерфейсу буде помітно. Це де HTML5 Web Sockets прийшов в руці. У нас було кілька побоювання з приводу результату, але це насправді перевершив всі наші очікування. Додаток виявилося бути досить швидко, щоб відправити джерело відеосигналу в якості вихідного бінарного зображення для кожного кадру без будь-яких помітних затримок. Отже, як ми застосували HTML5 для розробки користувальницького інтерфейсу, ми були не в змозі використати елементи управління WPF з Kinect SDK. З іншого боку, не обмежуючись WPF потужностей означає, що ми могли б вибрати альтернативний підхід тут. Наприклад, ми вибрали серветки рухатися замість захопити і перетягнути взаємодій.

Підтримка безшовної взаємодії

Для того, щоб додати більше інтерактивності в процесі голосування, ми вирішили поєднати частини установки за допомогою сервера статистики. Ідея полягає в тому, щоб з'єднати два або більше UIs до однієї системи, так що, наприклад, коли хтось "любить" проект в першій установці, інші частини стало відомо про це, і їх інтерфейс реагує відповідним чином. З технічної точки зору, для встановлення з'єднання, ми використовували ZeroMQ для зв'язку між вузлами, Pub / Sub каналу для повідомлення установок про будь-які зміни в статистиці і тягни / штовхай один із способів спілкування. В результаті, шляхом проб і помилок, нам вдалося зберегти безшовну взаємодію між призначеним для користувача інтерфейсом і драйвером Kinect, що має затримку знижується до майже непомітного значення.

## Концепція технічного рішення

Для реалізації рішення було обрано такі технології:

|  |  |
| --- | --- |
| Елемент | Технологія |
| Інтерфейс користувача | HTML5/JS |
|  | AngualrJS |
|  | WebSockets – коммунікація |
| Сервер | .NET |
|  | ZeroMQ – комунікація |
| Сенсор | Kinect |
|  |  |

Рішення виконано у вигляді гібридної хардварно-софтверної системи де користувач для керування взаємодіє через сенсор Kinect, а отримує інорфмацію через веб інтерфейс що за допомогою проектора виводиться на стіну. Інформація про з кожного сенсора обробляється за допомогою .НЕТ додатку і в разі необхідності передається на сервер статистики. Система побудована на принципах реативності і асинхронності. Коммінкація між компонентами максимально оптимізована.

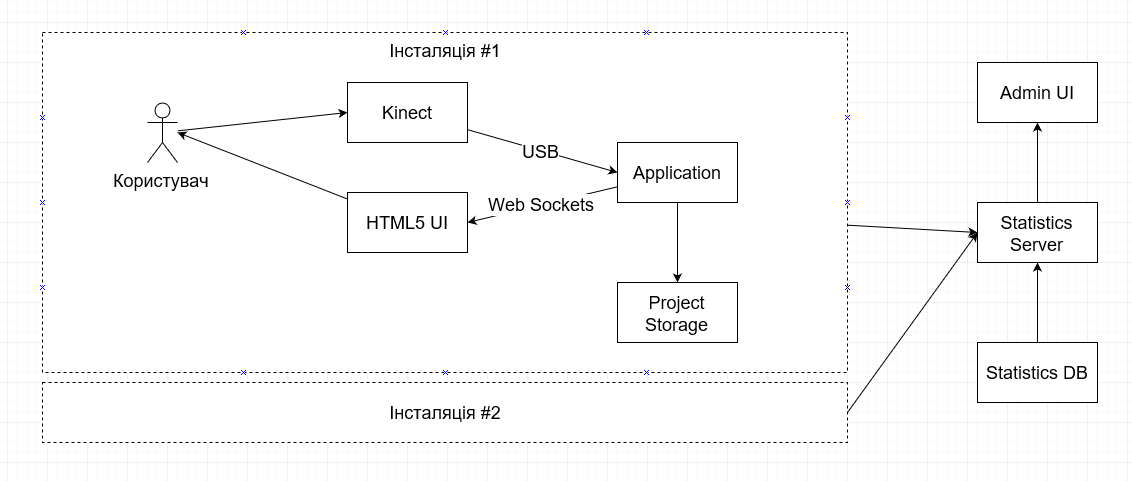


Рис. 4. Логічна схема рішення

## Концепція фізичного рішення

Фінальне рішення складається з сенсора Kinect під’єднанного до сервера, на якому йде обробка рухів користувачів, та інтерфейсу що реагує на рухи. За допомогою проектора інтерфейс користувача продукується на стіну або на великий екран. Приблизне розташування елементів інсталяції показано на діаграмі нижче.

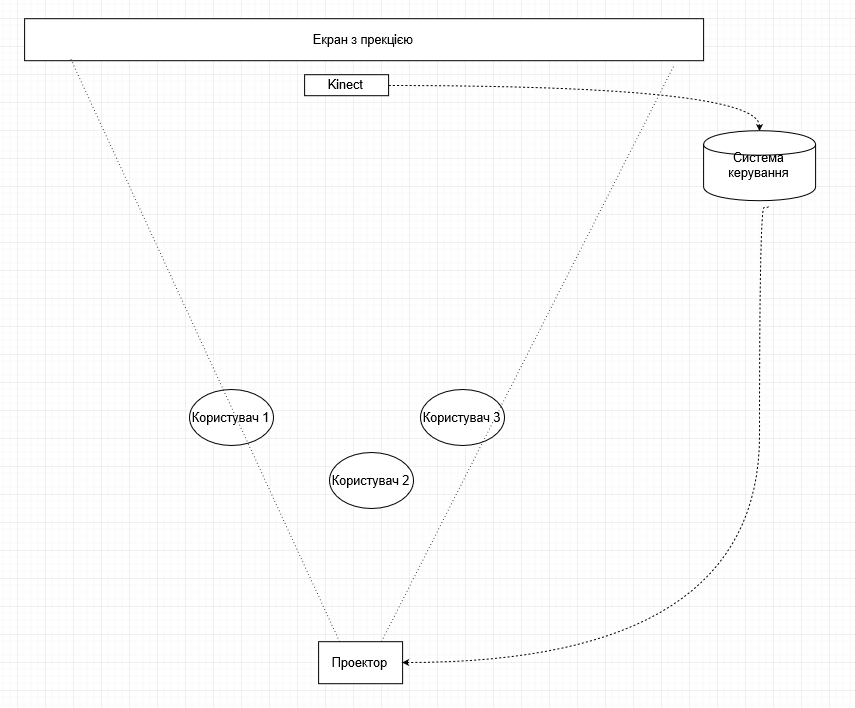


Рис. 5. Фізичне розташування компонент рішення

## Реалізація

Повний код проекту можна знайти в офіціальному репозиторії - <https://github.com/eleks/rnd-kinect-behance>

Тут наведені головні частини проекту:

#### Лістінг 1. Program.cs

﻿namespace Eleks.KinectBehance.KinectGlue

{

using Emitters;

using System;

using System.Reactive.Linq;

using System.Threading;

using Contracts.Events;

using Contracts.Commands;

class Program

{

static void Main(string[] args)

{

var console = new ConsoleManager(args);

var rnd = new Random();

var ws = new PresenterManager();

var kinect = new KinectManager();

using (var stats = new StatsServerManager(console.Server))

{

var heuristicsGueastures = new HeuristicsGuesturesEmitter(kinect.BodyReady);

heuristicsGueastures.GestureDetected.Subscribe(ws.Send);

heuristicsGueastures.GesturePointChanged.Subscribe(ws.Send);

//var vgbGestures = new VgbGesturesEmiter(kinect.KinectChanged, kinect.BodyReady);

//vgbGestures.GestureDetected.Subscribe(ws.Send);

//var starGestures = new StarGestureEmitter(kinect.BodyReady);

//starGestures.GestureDetected.Subscribe(ws.Send);

//starGestures.GestureDetected.Subscribe(Console.WriteLine);

var likeGestures = new LikeGestureEmitter(kinect.BodyReady);

likeGestures.GestureDetected.Subscribe(ws.Send);

likeGestures.GestureDetected.Subscribe(Console.WriteLine);

var position = new PositionAndSkeletonEmitter(kinect.BodiesReady);

position.PositionChanged.Subscribe(ws.Send);

position.SkeletonChanged.Subscribe(ws.Send);

var interaction = new InteractionStreamEmitter(kinect.BodyReady);

interaction.InteractionChanged.Subscribe(ws.Send);

var singlePersonTracking = new SinglePersonTracking(kinect.KinectChanged, kinect.BodiesReady);

singlePersonTracking.TrackingChanged.Subscribe(ws.Send);

singlePersonTracking.TrackingChanged.Subscribe(Console.WriteLine);

stats.EventReceived

.OfType<ProjectLikeCountChanged>()

.Subscribe(ws.Send);

ws.CommandRecieved.Subscribe(c =>

{

if (c is RaiseProjectLike)

{

Console.WriteLine(c);

var raiseCmd = (RaiseProjectLike)c;

var registerCmd = new RecordProjectLike

{

ClientId = console.ClientId,

ProjectId = raiseCmd.ProjectId,

Timestamp = DateTime.Now

};

stats.Send(registerCmd);

}

else

{

stats.Send(c);

}

});

stats.Start();

kinect.Start();

while (!console.IsCancelRequested)

{

Thread.Sleep(TimeSpan.FromSeconds(5));

}

// Close

kinect.Stop();

}

}

}

}

#### Лістінг2. PositionAndSkeletonEmitter.cs

﻿namespace Eleks.KinectBehance.KinectGlue.Emitters

{

using AppEvents;

using Microsoft.Kinect;

using System;

using System.Reactive.Subjects;

using System.Linq;

using System.Collections.Generic;

using Eleks.KinectBehance.Contracts.Events;

public class PositionAndSkeletonEmitter

{

readonly Subject<PositionChanged> positionChangedSubject = new Subject<PositionChanged>();

readonly Subject<SkeletonChanged> skeletonChangedSubject = new Subject<SkeletonChanged>();

readonly static ISet<JointType> JointsToTrack = new HashSet<JointType>

{

JointType.Head,

JointType.SpineShoulder,

JointType.HandLeft,

JointType.WristLeft,

JointType.ElbowLeft,

JointType.ShoulderLeft,

JointType.HandRight,

JointType.WristRight,

JointType.ElbowRight,

JointType.ShoulderRight

};

public PositionAndSkeletonEmitter(IObservable<BodiesReady> bodiesReady)

{

bodiesReady.Subscribe(e => OnBodiesReady(e.Bodies));

}

public IObservable<PositionChanged> PositionChanged

{

get

{

return positionChangedSubject;

}

}

public IObservable<SkeletonChanged> SkeletonChanged

{

get

{

return skeletonChangedSubject;

}

}

void OnBodiesReady(Body[] bodies)

{

foreach(var body in bodies)

{

if (!body.IsTracked)

{

continue;

}

positionChangedSubject.OnNext(new PositionChanged

{

Position = body.Joints[JointType.SpineShoulder].Position

});

var skeletonJoints = body.Joints.Values

.Where(x => JointsToTrack.Contains(x.JointType))

.Select(x => new SkeletonChanged.SkeletonJoint

{

JointType = x.JointType,

Position = x.Position,

TrackingState = x.TrackingState

})

.ToArray();

skeletonChangedSubject.OnNext(new SkeletonChanged

{

Joints = skeletonJoints

});

}

}

}

}

﻿Лістінг SwipeHorizontalGestureDetector.cs

﻿namespace Eleks.KinectBehance.KinectGlue.Emitters.HeuristicGestures

{

using Microsoft.Kinect;

using System;

using System.Collections.Generic;

using System.Linq;

using Utils;

public class SwipeHorizontalGestureDetector : IHeuristicGestureDetector

{

public float SwipeMinimalX { get; set; }

public float SwipeMaximalY { get; set; }

public int SwipeMininalDuration { get; set; }

public int SwipeMaximalDuration { get; set; }

public float SwipeMaximalZ { get; set; }

public SwipeHorizontalGestureDetector()

{

SwipeMinimalX = 0.7f;

SwipeMaximalY = 0.3f;

SwipeMininalDuration = 250;

SwipeMaximalDuration = 1500;

SwipeMaximalZ = 0.4f;

}

public string DetectGesture(IEnumerable<GestureEntry> entries)

{

if (entries.Max(x => x.Position.X) - entries.Min(x => x.Position.X) < SwipeMinimalX)

{

// Outside of horizontal window

return null;

}

if (entries.Max(x => x.Position.Y) - entries.Min(x => x.Position.Y) > SwipeMaximalY)

{

// Not enoyugh Y

return null;

}

//if (entries.Max(x => x.Position.Z) - entries.Min(x => x.Position.Z) > SwipeMaximalZ)

//{

// // Too big Z

// return null;

//}

var totalMilliseconds = (entries.Last().Time - entries.First().Time).TotalMilliseconds;

if (totalMilliseconds >= SwipeMaximalDuration && totalMilliseconds <= SwipeMininalDuration)

{

// Too slow or too fast

return null;

}

if (entries.Segments().All(x => x.Item2.Position.X - x.Item1.Position.X > -0.01f))

{

Console.WriteLine("SwipeToRight:" + entries.Count());

// Points should go to the up direction

return "SwipeToRight";

}

if (entries.Segments().All(x => x.Item2.Position.X - x.Item1.Position.X < 0.01f))

{

Console.WriteLine("SwipeToLeft:" + entries.Count());

// Points should go to the up direction

return "SwipeToLeft";

}

return null;

}

}

}

# Список використаних джерел

1. Документація Kinect SDK - <https://developer.microsoft.com/en-us/windows/kinect>
2. Кожевников, Г. К.; Трохимчук, С. Н.; Безникова, Е. В. Информационная Система «ПРОФКОМ ВУЗ». Вестник Харьковского национального автомобильно-дорожного университета, 2010.
3. Збірник нормативних документів Національного університету “Львівська політехніка” / За ред. професора Ю.Я. Бобала. – Львів: Видавництво Львівської політехніки, 2012. – 772 с.