УДК 004.428.4
Ящук А.А., Саварин П.В. Корнійчук Н.І.
Луцький національний технічний університет

СИСТЕМА ВІЗУАЛІЗАЦІЇ І ПЕРЕГЛЯДУ 3D МОДЕЛЕЙ НА ОСНОВІ БІБЛІОТЕК HELIX TOOLKIT І EMGU CV

Ящук А.А., Саварин П.В., Корнійчук Н.І. Система візуалізації і перегляду 3D моделей на основі бібліотек Helix Toolkit і Emgu CV. У статті розглянуто створення та дослідження ефективності системи, що дозволяє за допомогою бібліотеки Helix Toolkit візуалізувати тривимірні моделі об'єктів, за допомогою веб-камери і бібліотеки Emgu CV розпізнавати обличчя користувача і адаптувати відображення 3D моделі на екрані комп'ютера залежно від положення обличчя в режимі реального часу.
Ключові слова: 3D модель, розпізнавання образів, Helix Toolkit, Emgu CV, C#, .NET Framework, web-камера.

Ящук А.А., Саварин П.В., Корнийчук Н.И. Система визуализации и просмотра 3D моделей на основе библиотек Helix Toolkit и Emgu CV. В статье рассмотрено создание и исследование эффективности системы, позволяющей с помощью библиотеки Helix Toolkit визуализировать трехмерные модели объектов, с помощью веб-камеры и библиотеки Emgu CV распознавать лицо пользователя и адаптировать отображения 3D модели на экране компьютера в зависимости от положения лица в режиме реального времени.
Ключевые слова: 3D модель, распознавание образов, Helix Toolkit, Emgu CV, C#, .NET Framework, web-камера.

Yashchuk A.A., Savaryn P.V., Korniychuk N.I. The system for visualization and viewing of 3D models based on the Helix Toolkit and Emgu CV libraries. The development and research of efficiency of the computer system, which allows visualizing 3D object models using the Helix Toolkit library, recognizing the user's face using the webcam and library Emgu CV and adapt the display of 3D models on the computer screen depending on the position of the person in real time mode.
Key words: 3D model, pattern recognition, Helix Toolkit, Emgu CV, C#, .NET Framework, webcam.

Постановка проблеми. Технології візуалізації тривимірних об'єктів знаходять широке застосування в різних галузях, зокрема в системах комп'ютерного проектування, при моделюванні процесів, в індустрії відеоігор. Можливості відображення 3D моделей на двовимірних екранах обмежені. Зокрема зміна положення спостерігача не впливає на ракурс зображення 3D моделі на екрані, на відміну від об'ємних об'єктів реального світу. Сучасні технології розпізнавання образів дозволяють виявляти обличчя у відео потоці з високою точністю. В результаті аналізу інформації з web-камери, можна визначити розташування обличчя користувача відносно екрану, на якому відображається 3D модель, на основі чого в режимі реального часу оновлювати ракурс 3D моделі подібно до того, як це відбувається з об'єктами реального світу. Це дозволить зробити перегляд тривимірних об'єктів на двовимірних екранах більш природнім і реалістичним.

Аналіз останніх досліджень і публікацій. Для розпізнавання образів в режимі реального часу широкого застосування набув метод Віоли-Джонса (Viola and Jones) [1] на основі ознак Хаара. У роботі Роберта Куіма (Robert Kooima) [2] описується модель узагальненої проекції перспективи, що може бути використана при відображенні тривимірних об'єктів на екрані.

Мета. Метою є розробка та дослідження ефективності роботи системи візуалізації і перегляду 3D моделей на екрані з можливістю відстежування положення обличчя користувача в режимі реального часу і відповідно до цього зміни ракурсу тривимірної моделі із застосуванням бібліотек Helix Toolkit і Emgu CV.

Виклад основного матеріалу і обґрунтування результатів досліджень. Для реалізації системи використовувалось інтегроване середовище розробки програмного середовища Visual Studio 2013 і мова програмування C#. Відображення 3D об'єктів в розробленій системі реалізовано з застосуванням бібліотеки Helix Toolkit.

Підтримуваними форматами файлів 3D моделей є: .stl і .obj, що забезпечується засобами бібліотеки Helix Toolkit. При завантаженні модель масштабується до розміру вікна програми задопомогою методу FitSize():

```
void FitSize(int k)
        {
            double maxBound = Math.Max(Math.Max(
            device3D.Transform.TransformBounds(device3D.Content.Bounds).SizeX
            ,
```

```
            device3D.Transform.TransformBounds(device3D.Content.Bounds).SizeY
            ),
            device3D.Transform.TransformBounds(device3D.Content.Bounds).SizeZ
      );
            double scaleCoefficient = 100 / maxBound;
            var size = new Vector3D(scaleCoefficient, scaleCoefficient,
scaleCoefficient);
            var matrix = device3D.Transform.Value;
            matrix.Scale(size);
            device3D.Transform = new MatrixTransform3D(matrix);
            device3D.Transform.TransformBounds(device3D.Content.Bounds);
            viewPort3d.Camera.Position = new Point3D(k, 0, 0);
            viewPort3d.Camera.LookDirection = new Vector3D(-k, 0, 0);
            viewPort3d.Camera.UpDirection = new Vector3D(0, 0, 1);
            viewPort3d.CameraController.CameraTarget = new Point3D(0, 0, 0);
        }
```

Розпізнавання облич реалізовано з застосуванням бібліотеки Emgu CV на основі ознак Хаара [1]. Метод для виявлення обличчя виглядає наступним чином:

```
public static void Detect(IInputArray image, String faceFileName,
List<Rectangle> faces, out long detectTime)
        {
            Stopwatch watch;
            using (InputArray iaImage = image.GetInputArray())
            {
                    //Прочитати об'єкти каскаду Хаара
                    using (CascadeClassifier face = new
CascadeClassifier(faceFileName))
                    {
                        watch = Stopwatch.StartNew();

                        using (UMat ugray = new UMat())
                        {
                            CvInvoke.CvtColor(image, ugray,
Emgu.CV.CvEnum.ColorConversion.Bgr2Gray);
//нормалізація яскравості і збільшення контрасту зображення
                            CvInvoke.EqualizeHist(ugray, ugray);
//виявити обличчя на сірому зображенні і зберегти координати як прямокутник
                            Rectangle[] detectedFaces =
face.DetectMultiScale(
                               ugray,1.1,10,new Size(100, 100));
                            faces.AddRange(detectedFaces);
                        }
                        watch.Stop();
                    }
                detectTime = watch.ElapsedMilliseconds;
            }
        }
```

Початковими параметрами для визначення ракурсу 3D сцени є координати *x* і *y* центру виявленого обличчя, а також геометричні розміри контуру виявленого обличчя в пікселях zoom (усереднене значення ширини і висоти). Для зміни ракурсу 3D сцени використовувалися методи *FitView()* і *LookAt()* з бібліотеки Helix Toolkit:

```
    public void FitView(Vector3D newDirection, Vector3D newUpDirection,
double animationTime = 0);

    public static void LookAt(this ProjectionCamera camera, Point3D target,
Vector3D newLookDirection, Vector3D newUpDirection, double animationTime);
```

Тоді метод, призначений для зміни ракурсу 3D сцени, з відповідно до значення змінних *x*, *y* і *zoom* в даний момент часу буде мати наступний вигляд:

```
private void Look(double x, double y, double zoom, double k1, double k2)
{
            // обертання 3D моделі
            viewPort3d.FitView(new
Vector3D(viewPort3d.Camera.LookDirection.X, -x + k1, y – k2),
viewPort3d.Camera.UpDirection);

            // зміщення 3D моделі
            if (RotateOnlyCheckBox.IsChecked == false)
            {
                viewPort3d.Camera.LookAt(new
Point3D(viewPort3d.CameraController.CameraTarget.X, x – k1, -y + k2),
viewPort3d.Camera.LookDirection, viewPort3d.Camera.UpDirection, 0);
            }

            // зближення/віддалення моделі
            viewPort3d.Camera.LookAt(new
Point3D(viewPort3d.CameraController.CameraTarget.X - zoom,
viewPort3d.CameraController.CameraTarget.Y,
viewPort3d.CameraController.CameraTarget.Z), viewPort3d.Camera.LookDirection,
viewPort3d.Camera.UpDirection, 0);
}
```

Поправочні коефіцієнти *k1* і *k2* призначені для калібрування і центрування 3D моделі залежно від початкового положення виявленого обличчя.

Для збільшення ефекту глибини зображення сцени реалізовано зміщення картинки заднього фону сцени при зміні положення обличчя спостерігача. Метод для зміщення має наступний вигляд:

```
 Matrix m;
   // метод для зміщення заднього фону
   void shiftBackground(double x1, double y1, double zoom1)
   {
       m = new Matrix();
       m.Translate(x1, y1);
       m.Scale(zoom1, zoom1);
       viewPort3d.Background.RelativeTransform = new MatrixTransform(m);
   }
```

Змінні *x1*, *y1* і *zoom1* є функціями від координат *x*, *y* і *zoom*, що визначають параметри контуру виявленого на зображенні обличчя.

Методи *look()* та *shiftBackground()* викликаються кожного разу при оновленні зображення з web-камери і виявлення обличчя.

Для усунення ефекту тремтіння зображення 3D сцени застосовано згладжування координат виявленого обличчя методом простої ковзної середньої [3]:

$$SMA_t = \frac{1}{n}\sum_{i=0}^{n-1} p_{t-1} = \frac{p_t + p_{t-1} + \dots + p_{t-i} + \dots + p_{t-n+2} + p_{t-n+1}}{n}, \quad (1)$$

де SMA_t – значення простого ковзної середнього в т. *t*;

n – кількість значень вихідної функції для розрахунку змінного середнього (згладжуючий інтервал);

p_{t-1} – значення вихідної функції в точці.

Отримане значення простий ковзної середньої відноситься до середини обраного інтервалу, однак, традиційно його відносять до останньої точки інтервалу.

Оптимальним інтервалом для згладжування визначено n=5 останніх показників координати виявленого обличчя в часовому ряді (рис.1).

В результаті тестування системи на комп'ютері з графічним адаптером Intel HD Graphics 4600, при роздільній здатності відео зображення 1280x720px мінімальна частота оновлення 3D сцени склала 29 кадрів за секунду, середня частота – 62 кадри за секунду для тривимірних об'єктів з високою деталізацією. Загальний вигляд системи показано на рис.2.

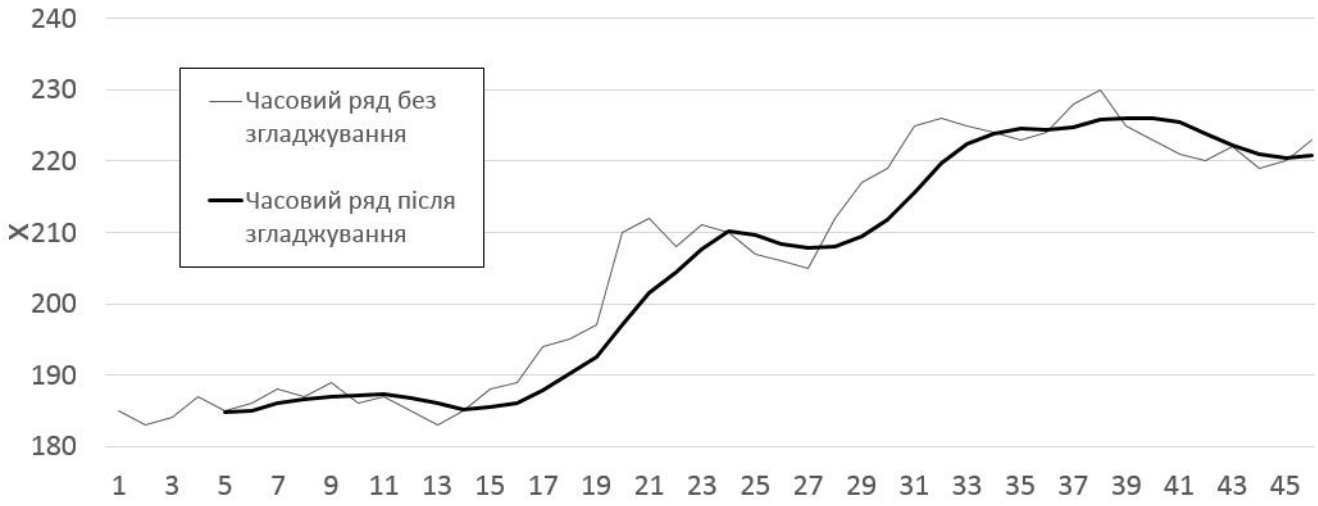

Рис.1. Згладжування часового ряду координати *X* методом простої ковзної середньої останніх 5 показників

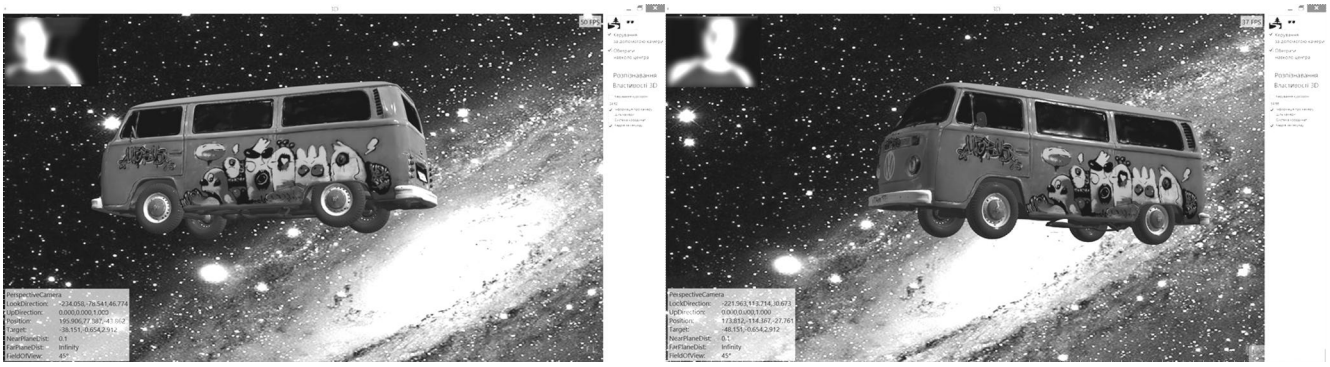

Рис. 2. Робота системи візуалізації і перегляду 3D моделей зі зміною ракурсу 3D сцени, залежно від розташування обличчя спостерігача

При функціонуванні даної системи потрібно враховувати ряд обмежень:

- система не може адаптувати відображення 3D моделі для більше, ніж одного користувача одночасно. Використовуючи технологію розпізнавання облич можна встановити пріоритетне обличчя особи, для якої система адаптуватиме проекцію 3D моделі, ігноруючи інші обличчя в кадрі;
- якість зображення з web-камери знижується при недостатньому освітленні, що негативно впливає на виявлення облич, а відповідно і на функціонування системи. Використання web-камери з вищою світлочутливістю дозволяє зменшити негативний вплив цього фактора.
- функціональність системи обмежується областю видимості камери. Використання web-камери з більшою роздільною здатністю і кутом огляду дозволяє зменшити негативний вплив цього фактора.
- ракурс зображення для обох очей – однаковий, на відміну від перегляду об'єктів реального світу, що негативно впливає на сприйняття глибини зображення. Для вирішення даної системи необхідні додаткові технічні засоби, що дозволяють переглядати стереозображення, а також адаптація розробленої системи для роботи з ними.

Висновки. Розроблено систему візуалізації і перегляду 3D моделей на екрані, яка працює на базі платформи .NET framework. Для відображення тривимірних об'єктів використано бібліотеку Helix Toolkit. Особливістю розробленої системи є відстежування в режимі реального часу положення обличчя користувача відносно екрану, на якому відображається 3D модель зі зміною ракурсу 3D сцени відповідно до цього. Виявлення обличчя користувача відбувається за допомогою бібліотеки Emgu CV у відеопотоці, що надходить з web-камери. В результаті тестування розробленої системи підтверджено її ефективність, зокрема високу продуктивність і стабільність системи, більш реалістичне сприйняття 3D об'єктів користувачем в порівнянні з переглядом зі статичним ракурсом.

1. Viola P. Robust real-time face detection. P. Viola and M. Jones / IJCV. – 57 (2). – 2004.
2. Robert Kooima. Generalized Perspective Projection August 2008, revised June 2009 [Електронний ресурс] / Режим доступу: http://csc.lsu.edu/~kooima/pdfs/gen-perspective.pdf – Назва з екрану.
3. Shumway R.H. Time Series Analysis and Its Applications: With R Examples. Third Edition R.H. Shumway and D.S. Stoffer / Springer Science+Business Media, LLC. – 2011. – 591 p.
4. Мишулина О. А. Статистический анализ и обработка временных рядов. – О. А. Мишулина / М.: МИФИ, 2004. – С. 180. – ISBN 5-7262-0536-7
5. Начало работы с OpenCV и его применение в C# [Електронний ресурс] / Режим доступу: https://habrahabr.ru/post/260741/ – Назва з екрану.
6. Helix Toolkit's documentation [Електронний ресурс] / Режим доступу: http://docs.helix-toolkit.org/en/latest/index.html – Назва з екрану.
7. Emgu CV Library Documentation [Електронний ресурс] / Режим доступу: http://www.emgu.com/wiki/files/3.3.0/document/html/8dee1f02-8c8a-4e37-87f4-05e10c39f27d.htm – Назва з екрану.
8. C# programming guide [Електронний ресурс] / Режим доступу: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/ – Назва з екрану.