Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего образования  
«Алтайский государственный технический университет им. И.И. Ползунова» Факультет информационных технологий  
Кафедра "Прикладная математика"

|  |
| --- |
| Курсовой проект защищен с оценкой в |
| Преподаватель Е.Н. Крючкова  Подпись |
| « » 2019 г. |

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ  
Проектирование Фреймворка для системы**

**трекинга объектов**

по дисциплине «Архитектурное проектирование и паттерны программирования»

**КП 09.03.04.47.00 ПЗ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент группы ПИ-52 Р.Я. Халилов | | | | |
|  |  |  | подпись | и.о.,фамилия |
| Преподаватель к.ф. -м.н, доцент Е.Н. Крючкова | | | | |
| должность, ученая степень подпись | | | | и.о.,фамилия |

БАРНАУЛ 2019

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Задание**  Учебная дисциплина: Архитектурное проектирование и паттерны программирования  ФИО студента: Халилов Рустам Яшарович  Группа: ПИ-52  Тема курсового проекта: Проектирование Фреймворка для системы  трекинга объектов  Этапы разработки расчетного задания и сроки их выполнения:   1. Анализ проблемы и существующих в мировой практике методов решения поставленной задачи (31.01.2018 – 31.03.2018) 2. Выбранные алгоритмы и обоснование выбора (16.04.2018 – 07.05.2018) 3. Паттерны программирования, применяемые в проектируемой системе и цели их использования (07.05.2018 – 11.05.2018)   4. Архитектура системы, диаграмма классов (09.04.2018 – 21.05.2018)   1. Перспективы развития системы (21.05.2018 – 25.05.2018); 2. Сдача работы руководителю и защита работы (25.05.2018 – 28.05.2018);   Дата выдачи задания: 26.01.2019 Срок защиты: 26.01.2019  Руководитель: доцент П.И. Ананьев | | | | | | | | | | |
|  |  |  |  |  | КП 09.03.04.47.000 ПЗ | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № докум. | Подп. | Дата |
| Разраб. | | Р.Я. Халилов |  |  | Проектирование Фреймворка для системы трекинга объектов | Лит. | | | Лист | Листов |
| Пров. | | Е.Н.Крючкова |  |  |  | У |  | 2 | 24 |
|  | |  |  |  | АлтГТУ, ФИТ  *гр. ПИ-52* | | | | |
| Н.контр. | | Е.Н.Крючкова . |  |  |
| Утв. | | Кантор С.А |  |  |

**Аннотация**

В курсовом проекте была спроектирован и реализован фреймворк системы трекинга объектов. Проектирование проекта было осуществлено с применением паттернов программирования и в дальнейшем реализовано в проекте C# с графическим интерфейсом WPF.

Фреймворк позволяет идентифицировать объекты по заранее заданным шаблонам и выделять их удобным пользователю способом.

Архитектура проекта фреймворка и приложение С# записаны на прилагаемом к данному отчету диске.

**Содержание**

Введение.................................................................................................5

1. Анализ проблемы и существующих в мировой практике

методов решения поставленной задачи..........................................6

2. Выбранные алгоритмы и обоснование выбора............................13

3. Паттерны программирования, применяемые в проектируемой  
 системе и цели их использования..................................................14

4. Архитектура системы в целом, диаграмма классов.....................15

5. Перспективы развития системы.....................................................16

6. Проведенный вычислительный эксперимент,

примеры работы................................................................................17

Заключение............................................................................................19

Список литературы...............................................................................20

Приложение А. Код программы..........................................................21

**Введение**

В связи с развитием искусственного интеллекта, в частности систем распознавания образов и ростом вычислительных мощностей, трекинг объектов стал использоваться повсеместно. Он затрагивает множество различных областей и используется в системах обнаружения нарушений на дорогах, в авиации. Наибольшее распространение трекинг приобрел в области видеомонтажа для операций с траекторией объекта.

Как правило трекинг подразумевает наличие маркера, которым может выступать датчик или область изображения и дальнейший анализ изображения с целью распознавания маркеров и отслеживания их новой позиции.

Развитие систем трекинга объектов является актуальной, поскольку все чаще и чаще используется в нашей жизни для решения сложных задач самых разных направлений.

1. **Анализ проблемы и существующих в мировой практике методов решения**

В данное время существует довольно много алгоритмов трекинга объектов в реальном времени, использование которых практически не сказывается на производительности видеопотока, за счет низкой вычислительной сложности этих алгоритмов.

Характеристики целевого объекта в реальном видео с течением времени, как правило, изменяются в силу естественных условий видеосъемки, когда, например, во время съемки изменяется освещенность объекта, изменяются относительные размеры объекта в последовательности кадров при удалении/приближении объекта относительно камеры, объект слежения заслоняется другими объектами и т. п.

В этих условиях практическое применение алгоритмов трекинга приводит к следующим ошибкам:

* Смещение окна выделяемого объекта
* Потеря объекта
* Фокус подобного объекта, вместо нужного

Проблема уменьшение количества ошибок и повышение стабильности работы является одной из проблем трекинга объектов.

Общий алгоритм трекера выглядит следующим образом:

На входе имеется некоторый кадр изображения. Пометим маркером на изображении некоторый объект который необходимо отследить. Маркер содержит все данные необходимые признаки объекта такие как: яркостная характеристика, цветовая характеристика, гистограмма и т.д. Признаки могут отличатся в зависимости от конкретного алгоритма.

Если объект помечен маркером, на следующем кадре определяется положение в которое переместился объект. Во многих алгоритмах трекинга новое положение объекта на данном кадре находится с помощью функций, позволяющих сравнивать характеристики различных областей, выделенных на кадрах, и находить область данного кадра с характеристиками, максимально схожими с характером области кадра предыдущего изображения.

Процент ложный срабатываний трекера или потери объекта растет если объект движется быстро, относительно частоты кадров, иначе говоря, в соседних кадрах наблюдается значительно перемещение объекта.

Для разных задач, используются свои модели движения, описывающие возможные изменения положения объекта:

* Плоский объект – расположен в 2D пространстве и его положение определяется с помощью двух координат
* Трехмерный объект – по мимо определения трех координат, иногда для таких объектов может требоваться еще и определение его ориентации в пространстве, что усложняет задачу трекинга.
* Модель для сжатия видео – ключевые кадры изображения разделяются на блоки, где каждый блок преобразуется при помощи вектора движения из параметров движения.
* Сеточное изображение – движение объекта определяется относительно вершин сетки.

**Детектирование:**

* 1. **Обученный классификатор**

Классификатор позволяет сравнивать объекты и выдавать заключение о пренадлежности объекта к классу с заданной вероятностью. Однако, для начала его необходимо обучить. Общая суть заключается в подаче на вход некоторый выборки объектов, класс которых уже определен. Классификатор накапливает общие параметры для данного класса и в дальнейшем на этой основе определяет принадлежит ли предполагаемый класс к заданному.

Для подобных алгоритмов, качество распознавания объектов пропорционально зависит от количества выборки.

* 1. **Сегментация**

Сегментация - это разбиение изображения на множество покрывающих его областей. Подобный алгоритм работает как функция от двух переменных F = L(x, y) где x, y – координаты пикселя. Значением функции может быть интенсивность или модуль градиента. Для данный функции определены локальные максимумы, разность которых с локальными минимумами определяет величину перепада интенсивность. В зависимости от допуска, в результате будет выделены максимумы, которые являются граничными точками некоторого объекта.

* 1. **Вычитание фона**

Зачастую требуется отслеживать на изображении множество объектов. Самым распространенным способом выделить объекты является метод вычитания фона. Он используется только когда камера закреплена статически и изображение меняется незначительно с течением времени. Смысл метода заключается в сравнении текущего изображения с изображением, которое заведомого не имеют объектов. Медианная разница в интенсивности цветов между фоновым и входным изображением показывает зону где находится объект на текущем кадре. Уменьшить количество ложных срабатываний можно если заранее известно количество объектов в кадре. Это применяется например для отслеживания перемещений лабораторных мышей в клетке с целью наблюдения за их социальным поведением.

* 1. **Kernel-based алгоритм**

Алгоритм представляет собой итеративную процедуру локализации объекта, основанная на максимизации критерия подобия.

Как правило данные представляются в виде вектора признаков объекта, для которых выбирается цветовая гистограмма слежения.

Рассмотрим основные из них:

* 1. **Алгоритм Mean-shift**

1. Пусть объект характеризуется некоторой гистограммой, а его положение на предыдущем кадре y.
2. Для нахождения объекта на текущем кадре вычисляется мера различия между гистограммой кандидата объекта p(y) и гистограммой объекта q
3. Вычисляем гистограмму окна для нового положения объекта и оцениваем схожесть гистограмм нового кандидата объекта и объекта
4. Шаг номер 3 повторяет до тех пор пока не достигнута приемлемая схожесть гистограмм.

Получившаяся область будет являться объектом в новом положении в пространстве

При неправильном выборе констант в формулах алгоритма может наблюдаться значительное количество ложных срабатываний, которые могут скачкообразно возрастать после некоторых значений.

* 1. **Optical Flow**

Метод оптического потока основан на изменении положения объекта относительно фона при параллаксе. В этом случае используются несколько входных изображений(камер) для съемки объекта с разных углов. Причем рабочей зоной будет считаться область пересечения камер. Метод позволяет получить трехмерные координаты и используется для виртуальной реальности, однако для корректной работы необходимо предварительно откалибровать камеры, при увеличении количества которых, сложность калибровки возрастает, однако улучшается точность позиционирования.

Дифференциальные методы оценки оптического потока, основанные на частных производных сигнала:

* Алгоритм Лукаса — Канаде — рассматриваются части изображения и аффинная модель движения
* Horn–Schunck — минимизация функционала, описывающего отклонение от предположения о постоянстве яркости и гладкость получаемого векторного поля.
* Buxton–Buxton — основан на модели движения границ объектов в последовательности изображений
* Общие вариационные методы — модификации метода Horn-Schunck, использующие другие ограничения на данные и другие ограничения на гладкость.
* Дискретные методы оптимизации — поисковое пространство квантуется, затем каждому пикселю изображения ставится в соответствие метка таким образом, чтобы расстояние между последовательными кадрами было минимальным. Оптимальное решение часто ищется с помощью алгоритмов нахождения минимального разреза и максимального потока в графе.

Трекинг при помощи оптического потока часто применяется при использовании стационарных камер, таких как камеры в аэропортах или зданиях, а также стационарные камеры-видеорегистраторы.

* 1. **Трекинг контуров**

Вариационный метод поиска границ в изображении. В описываемом

алгоритме используется модель активных контуров свободной формы. В этой модели задача поиска границы объекта формулируется как нахождение контура, на котором некоторый функционал достигает минимума. Модель имеет динамический характер (в процессе минимизации функционала контур меняется), отсюда название «активная». Поведение активного контура и его свойства полностью определяются функционалом E. Функционал контура зависит от формы и размеров контура и от его положения на изображении и записывается в виде суммы функционалов «внутренней» и «внешней» энергии контура.

Алгоритм выделяет границу объекта на изображении и анализирует найденную границу. На вход алгоритм получает изображение однотонного объекта, при этом для каждого объекта пользователь выставляет одну точку внутри этого объекта.

* 1. **Сравнение с шаблоном**

Технология для поиска областей на изображении, которые совпадают с изображением-шаблоном с заданной точностью.

Требуется два компонента – исходное изображение (например, снимок с камеры) и изображение-шаблон. Чтобы определить сходный участок, нам нужно наложить шаблон на снимок и попиксельно двигать его таким образом чтобы шаблон прошелся по всему изображению. В каждом положении вычисляется метрика, которая показывает, совпадают картинки или нет. Для каждого положения шаблона метрика заносится в результирующий массив.

После окончания сравнения находятся лучшие совпадения. И если совпадение было выше заданного порога, то оно считается данным объектом.

Данный метод является наиболее неточным. Его применение оправдано в случаях изображений, где объект перемещается в двумерной плоскости, не отдаляясь от камеры и соответственно не изменяя свои размеры.

Преимуществом данного метода является то, что он наиболее прост в реализации, однако, неустойчив.

1. **Выбранные алгоритмы и обоснование выбора**

Для реализации проекта фреймворка был выбран алгоритм сравнения с шаблоном, поскольку область применения фреймворка будет ограничена векторным двухмерным изображением с объектами, не изменяющими свою ориентацию и размеры. А поскольку векторное изображение не имеет сильных шумов, характерных для растровых изображений, например фотографий, сравнение шаблона с участками изображения можно будет провести за небольшое количество шагов, это позволит уменьшить количество входных данных для уменьшения времени распознавания объекта с минимальным влиянием на точность детектирования.

Программа будет получать на вход изображение, искать объекты, совпадающие с шаблоном, вокруг найденного объекта создавать рамку по запросу пользователя и генерировать слой изображения с наложенной рамкой.

1. **Паттерны программирования, применяемые в проектируемой системе и цели их использования**

При проектировании системы были использованы следующие паттерны:

1. **Adapter**

Паттерн используется для приведения интерфейса UI элемента к интерфейсу распознанного объекта. Поскольку интерфейс UI элемента предоставляет значительно больше информации которая не используется, однако требует дополнительных операций, в т.ч затратных по времени для доступа к свойствам, его адаптация под класс распознанного объекта была сделала в целях более быстрого и удобного доступа к свойствам и методам.

1. **Iterator**

Паттерн используется для последовательного доступа к известным шаблонам объектов для последующего их сравнения, без раскрытия внутренней структуры класса с набором шаблонов.

1. **Façade**

Скрывает реализацию детектирования объектов и предоставляет пользователю упрощенный интерфейс с возможностью настройки фона, надписи и рамки, выделяющей отслеживаемый объект.

1. **Information Expert**

Наделяет класс с набором шаблонов, обладающий максимумом информацией о шаблонах, функционалом редактирования, удаления и добавления шаблонов для повышения инкапсуляции и снижения уровня зацепления.

1. **Архитектура системы, диаграмма классов**

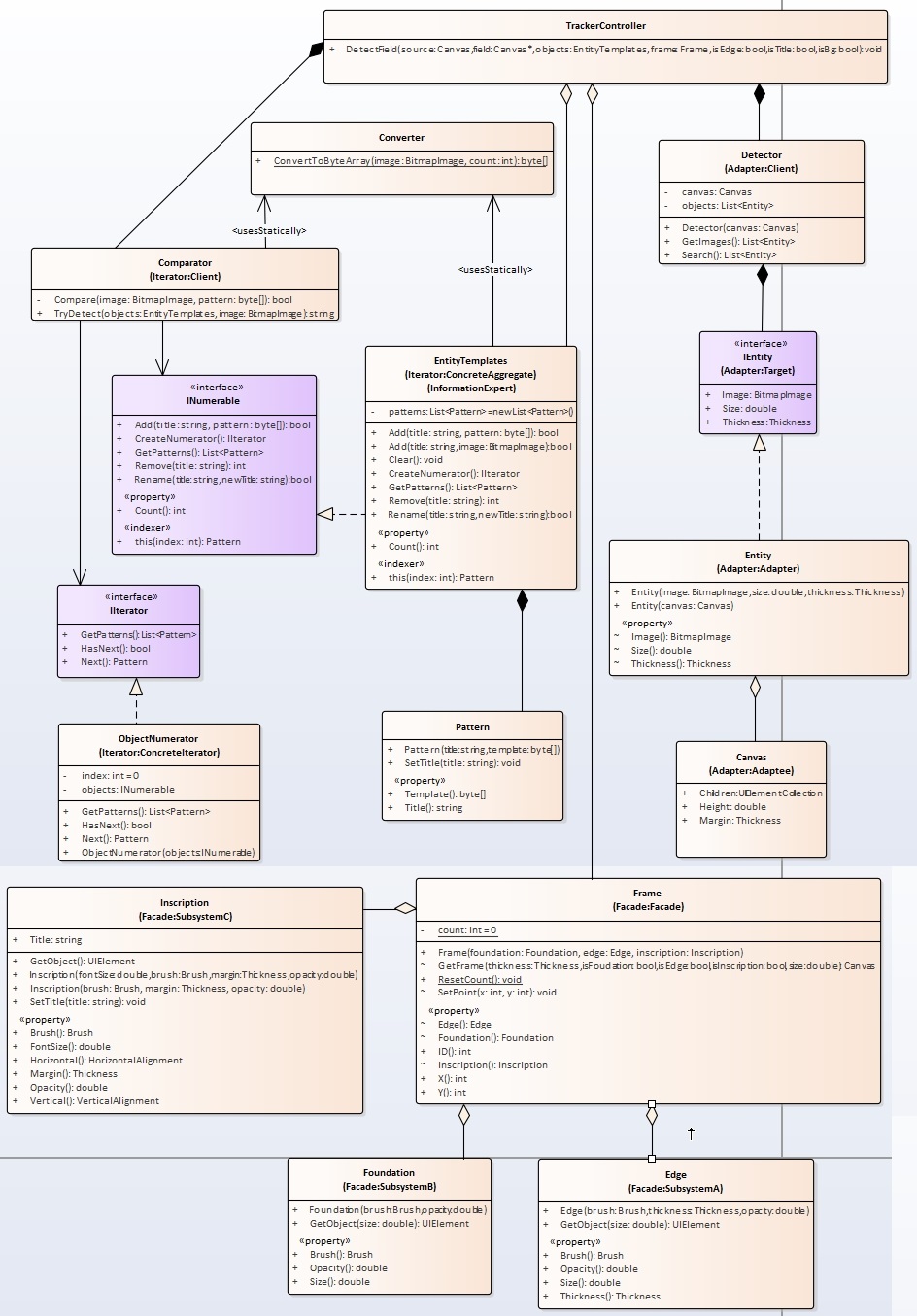


Рисунок 1. Диаграмма классов

1. **Перспективы развития системы**

В данную систему, помимо улучшения критических участков кода можно в дальнейшем расширить путем добавления функциональности отслеживания траектории с указанием положения объекта в заданное время и возможностью модифицировать эту визуальную часть для пользователя.

Так же возможно расширение путем добавления лог файлов, в которых будет содержаться время, траектория передвижения и класс объекта, который появился либо исчез с изображения.

1. **Проведенный вычислительный эксперимент, примеры работы**

Среднее время полной обработки кадра трекером занимает в среднем 30 мс для 20 объектов.

Пользователь задал рамку с синим фоном прозрачностью 20%, красным обрамлением толщиной 1 пиксель и белой надписью класса объекта в левом верхнем углу. Известные шаблоны – «Кот», «Сова», «Медведь».



Рисунок 2. Программа с 18 объектами

Пользователь задал рамку с красным фоном прозрачностью 20%, без обрамления и белой надписью класса объекта в левом верхнем углу. Известные шаблоны – «Утка», «Сова», «Курица», «Воробей»,

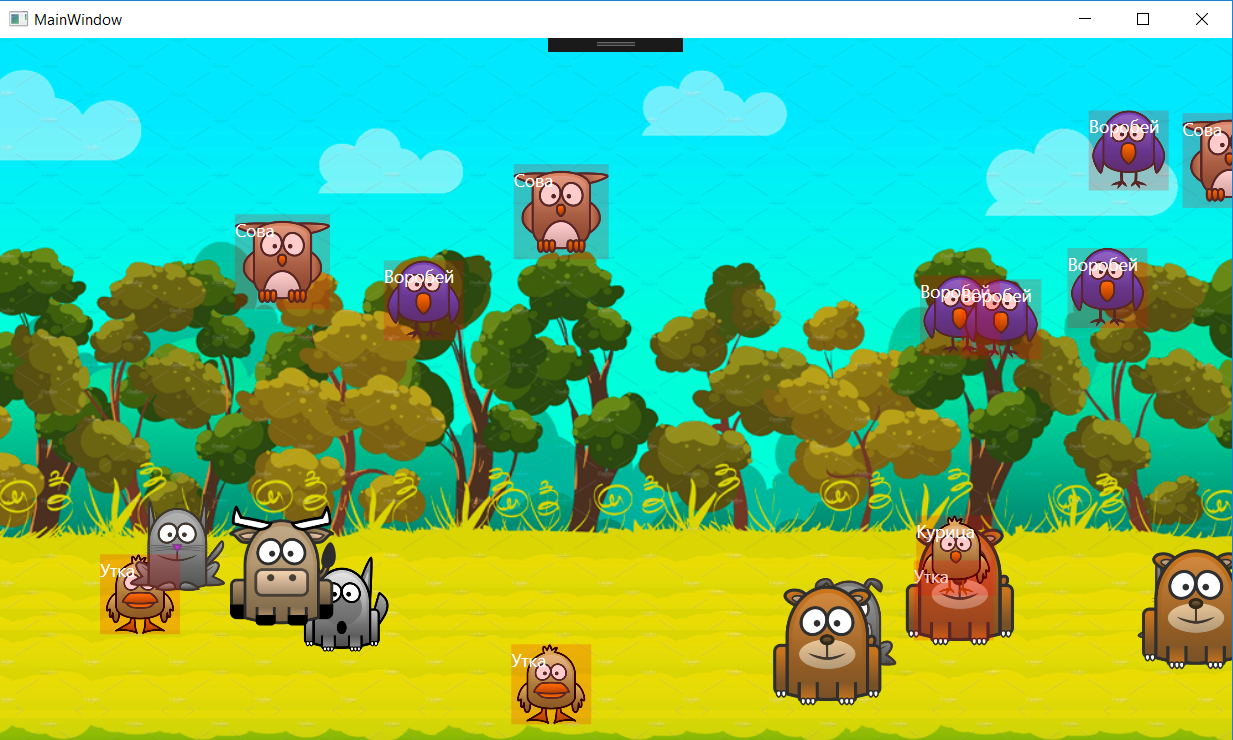


Рисунок 3. Программа с 20 объектами

**Заключение**

Таким образом, разработан фреймворк, позволяющий осуществлять трекинг объектов используя шаблоны, определять их класс и выделять их визуально.

Пользователю доступен функционал для настройки визуального выделения и операций с шаблонами объектов. Фреймворк удобен в использовании и скрывает сложный логический функционал от пользователя. Получена гибкая архитектура, которая соответствует принципам ООП, GRASP и содержит паттерны проектирования.

**Список использованной литературы**

1. Крючкова Е. Н., Старолетов С. М. Архитектурное проектирование и паттерны программирования: Учебно-методическое пособие.— Барнаул: АлтГТУ, 2015. – 109c.
2. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – Техносфера, 2005. – 1072 с.
3. Трекинг (компьютерная графика) [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Трекинг. – Заглавие с экрана. – (дата обращения: 25.01.2019).
4. Алгоритм трекинга объектов в реальном времени с обработкой ошибок [Электронный ресурс]. – Режим доступа: https://cyberleninka.ru/article/v/algoritm-trekinga-obektov-v-realnom-remeni-s-obrabotkoy-oshibok. – Заглавие с экрана. – (дата обращения: 25.01.2019).
5. Фасад (Facade) | Паттерны в C# и .NET [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/patterns/4.3.php. – Заглавие с экрана. – (дата обращения: 25.01.2019).
6. Итератор (Iterator) | Паттерны в C# и .NET [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/patterns/3.5.php. – Заглавие с экрана. – (дата обращения: 25.01.2019).
7. Обзор алгоритмов сегментации / Блог компании Intel / Хабр [Электронный ресурс]. – Режим доступа: https://habr.com/ru/company/intel/blog/266347/. – Заглавие с экрана. – (дата обращения: 25.01.2019).
8. Линейный классификатор [Электронный ресурс]. – Режим доступа: http://www.machinelearning.ru/wiki/index.php?title=Линейный\_классификатор. – Заглавие с экрана. – (дата обращения: 25.01.2019).
9. Отслеживание объектов в видеопотоке [Электронный ресурс]. – Режим доступа: https://sibac.info/studconf/tech/liv/80314. – Заглавие с экрана. – (дата обращения: 25.01.2019).
10. Canvas Class (System.Windows.Controls) | Microsoft Docs [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.controls.canvas?view=netframework-4.7.2. – Заглавие с экрана. – (дата обращения: 25.01.2019).

**Приложение А. Код программы**

Проект программы с исходным кодом находится на прилагаемом к отчету диске в папке «ObjTracking»