Санкт-Петербургский государственный университет

**Факультет прикладной математики – процессов управления**

**Кафедра технологии программирования**

Ригачный

Александр

Андреевич

Система вычисления положения видеокамеры для формирования виртуального оформления телевизионных спортивных трансляций

Заведующий кафедрой,

кандидат физ.-мат. наук,

доцент Сергеев С. Л.

Научный руководитель,

старший преподаватель Севрюков С. Ю.

Рецензент Лисеев С.К.

Санкт-Петербург

2009

**Оглавление**

[Введение 3](#_Toc224377113)

[Цели, задачи и возможности использования 6](#_Toc224377114)

[Постановка задачи](#_Toc224377115) 7

[Обзор общего алгоритма калибровки видеокамеры](#_Toc224377116) 8

[Алгоритмы предобработки изображения](#_Toc224377117) 9

Фильтрация изображения 9

Построение бинарного изображения маски поля 12

Поиск разметки игровой площадки на изображении 17

[Алгоритмы поиска прямых на бинарном изображении 20](#_Toc224377119)

Преобразование Хафа 21

Итеративно перевзвешиваемые наименьшие квадраты 24

[Алгоритм вычисления положения камеры по найденным прямым](#_Toc224377120) 28

Принцип формирования изображения в видеокамере 29

Вычисление матрицы гомографии 31

[Полученные результаты 3](#_Toc224377126)4

[Заключение](#_Toc224377127) 35

[Литература и ссылки](#_Toc224377133) 36

# Введение

Телевидение плотно вошло в нашу жизнь. Стремительное развитие телевидения во второй половине 20-ого века привело к тому, что уже выросло несколько поколений, не представляющих себе жизнь без телевизора. Качество вещания значительно возросло и стало цифровым. Телевидение стало нормой.

Если в советские годы телевидение было исключительно государственной прерогативой, то теперь появилось множество коммерческих телеканалов, каждый из которых борется за привлечение рекламодателей, которые приносят основную прибыль компании. Но для этого каналу необходимо, чтобы его смотрели. Это в свою очередь заставляет канал бороться и за зрителя. Рейтинг – основной показатель качества работы канала. Для повышения этого рейтинга канал обязан предоставлять программы качественные как в техническом, так и в художественном плане.

На сегодняшний день развитие телевизионных технологий достигло небывалого масштаба. На смену аналоговым системам, приходят цифровые. Основное преимущество цифрового телевидения заключается в улучшенном качестве изображения и звука. Стремительно развиваются цифровые форматы, и как следствие расширяется ассортимент соответствующего оборудования и сферы его применения.

В середине 2000-х годов началось стремительное развитие Телевидения Высокой Чёткости (ТВЧ). Сегодня ТВЧ – это самая современная и высокотехнологичная область цифрового телевидения, развитие которой подтверждает высокий потенциал страны в сфере информационных технологий и способствует скорейшему переходу телерадиовещания на цифровой формат.

Объёмы информации лавинообразно растут. Цифровое телевидение дает возможности “уплотнить” транспортные каналы, путем видеокомпрессии, тем самым позволяя на одном частотном канале, на котором раньше передавалась одна телевизионная программа, передавать сразу несколько.

Да и каналов доставки сейчас стало гораздо больше. Если раньше было лишь эфирное телевидение, то сейчас появилось спутниковое, кабельное, интернет телевидение (IP-TV), а также ведутся разработки по созданию вещания в 3G-сетях (мобильное телевидение).

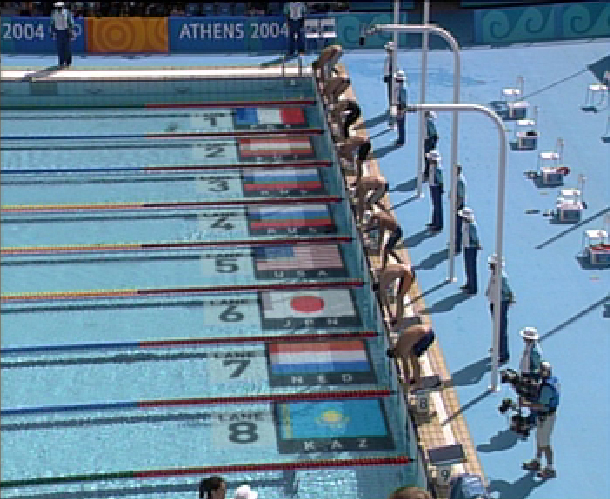
Помимо всего перечисленного, для повышения рейтинга, телеканалы используют внедрение интерактивного сервиса. Зритель получает возможность “общаться” с каналом средствами телефона и интернета. Люди могут, находясь дома перед экраном, участвовать в различных телевизионных играх, викторинах, голосованиях, sms-чатах, и даже управлять сценариями шоу и сериалов.

Невозможно представить какие новые способы привлечения зрителя будут применяться в телевидении уже через пару лет, но, безусловно, основными рейтинговыми жанрами останутся Новости, Кино и Спорт.

Бесспорно, именно трансляции спортивных событий играют огромную роль в привлечение аудитории. Футбол, хоккей, биатлон и теннис - одни из самых популярных видов спорта в России. В последние годы даже в футболе наша страна достигла многого. Мы являемся одной из самых спортивных держав в мире. Наши спортсмены регулярно увозят внушительные комплекты наград с каждой Олимпиады. В России проживает около 141 миллиона человек и большая часть из них внимательно следит за основными спортивными событиями в нашей стране и мире в целом, начиная от чемпионатов России по различным видам спорта и заканчивая чемпионатами Европы, Мира и Олимпиады.

Но, к сожалению, качество спортивных трансляций на наших телеканалах оставляет желать лучшего. В Европейских странах огромное внимание уделяется виртуальной графике. На видео изображение накладываются различные стрелки с расстояниями от той или иной точки, скорости игроков, высоту прыжков, эмблемы команд, флаги стран и многое другое, что, несомненно, привлекает зрителя (рис. 1 и 2).

Трансляции становятся более информативными и при этом не нагружают зрительное восприятие людей, потому как вся графика рисуется так, что кажется, будто она нарисована на площадке. Для рекламодателей, предоставляется возможность выводить рекламу прямо во время трансляции на заднем фоне или на площадке, создавая впечатление как будто это не виртуальная графика, а реальная.

**Рис. 1.** **Рис. 2.**

Для развития такой технологии в России был сформулирован проект формирования виртуальной графики для оформления спортивных телевизионных трансляций, целью которого является наложение различных логотипов и графики таким образом, чтобы она казалась визуально лежащей на игровой площадке или рядом с ней, в реальном времени.

Для осуществления такого наложения необходимо получать входящее с камеры видео изображение, вычислять позицию, куда рисовать графику, а затем рисовать саму графику. В данной работе не рассматриваются решение задач: получения кадров из видеопотока и вывода графики на экран. Основная цель – это как раз получение координат для графики. Но, чтобы вычислить необходимую позицию, нужно знать то положение камеры, в котором она находиться по отношению к полю. Для этого применяется два подхода: либо на камеру устанавливается специальный датчик, который следит за тем, как она повернута, либо же специальная программа анализирует входящее изображение и на основе разметки игрового поля или других статических предметов вычисляет позицию видеокамеры.

Первый метод не пользуется популярностью, так как телекомпания, транслирующая спортивное событие, не обязательно сама его снимает. Она может получать видео поток от другой телекомпании из другой страны, или просто показывать матч из архива, в таком случае у нее не будет возможности влиять на камеры, которыми снимается спортивное мероприятие.

Задачей моей работы как раз и является осуществление второго метода: Необходимо осуществить привязку видеокамеры к пространству на основе изображений получаемых с этой камеры.

Но каждый вид спорта настолько индивидуален, что невозможно создать единую систему оформления спортивных трансляций в эфире. Размеры площадок везде различны, как следствие различны положение и формы разметок. В данной работе пойдет речь о создании системы для самого популярного вида спорта в мире – футболе. Также в работе будут рассмотрены множество алгоритмов, которые можно успешно переносить на другие виды спорта, учитывая их особенности.

# Цели, задачи и возможности использования

***Основная цель работы***: написание системы анализа видео изображения, с целью поиска на нем спортивной площадки и определения её положения в пространстве относительно видео камеры на примере трансляции футбольного матча. Положение будет определяться путем обнаружения разметки поля на кадрах, так как других неподвижных предметов постоянно находящихся в объективе камеры попросту нет.

Для достижения этой цели в работе будут решены следующие задачи:

* Разбор различных методов и алгоритмов: фильтрации изображения, выявления областей связанности, поиска границ;
* Разбор методов поиска прямых на бинарном изображении. Обоснование выбора основного алгоритма поиска прямых на основе *итеративно перевзвешиваемых наименьших квадратов*;
* Разбор метода определения положения плоскости относительно камеры по найденным прямым на изображении. Вычисление гомографии и калибровка видеокамеры;

Рассмотренные в данной работе алгоритмы могут быть использованы в самых различных видах спорта с учетом их особенностей. Да и не обязательно в спорте: это могут быть и различные телешоу, и трансляции парадов. Результаты работы необходимы для решения многих задач в оформлении телевизионного эфира, таких как:

* Наложение видео графики на спортивную площадку;
* Обнаружение позиций игроков на поле, и позиций мяча;
* Подсчет различных игровых статистик, например: среднюю и максимальную скорости игроков, километраж, который они пробежали, процент владение мячом или процент нахождения мяча в той или иной зоне площадки, и другие.

# Постановка задачи

Имеется видео поток данных с трансляции футбольного матча. Поступающие изображения имеют формат RGB. Предполагается, что камера берет довольно крупный план, где видна хоть какая-то разметка поля. При этом камера может двигаться и совершать повороты, но движение её всегда плавно. В систему необходимо задать цвет игровой площадки. Это будет осуществляться пользователем этой программы при получении самого первого кадра. Также, пользователь должен будет сопоставить прямые, найденные системой на первом кадре, реальным линиям разметки поля. В дальнейшем система должна работать автономно, то есть влияние на неё человеком, происходит всего один раз в самом начале работы. Так же считаем, что первые несколько кадров камера неподвижна – это необходимо, чтобы система смогла первоначально как можно точно определить положение поля и его разметку. Когда цвет настроен и прямые сопоставлены, система начинает обрабатывать поступающие кадры с частотой 25 кадров в секунду. От системы требуется в последующих кадрах, вычислять новое положение площадки относительно движущейся камеры, зная предыдущее её положение. При этом она должна успевать обрабатывать каждое изображение за 40 милисекунд, для того чтобы выводить обработанные кадры в реальном времени, то есть с частотой 25 кадров в секунду. Следует отметить, что размеры футбольной площадки и её разметка нам известны.

# Обзор общего алгоритма калибровки видеокамеры

На входе имеется первое изображение. Системе необходимо определить положения площадки относительно камеры по этому изображению. Для этого на изображении будут искаться линии разметки футбольного поля. Вначале полученную картинку необходимо отфильтровать, для удаления из нее шумов. Затем, для того чтобы не искать разметку вне поля, система должна определить на кадре точки, относящиеся непосредственно к футбольному полю. Осуществляться это будет путем построения маски поля, то есть такого бинарного изображения, в котором черные пиксели обозначают точки, принадлежащие к игровой площадке, а белые - остальные точки кадра. После этого система будет рассматривать отфильтрованный кадр только в точках, относящихся непосредственно к полю. Среди этих точек, с помощью алгоритмов поисков краев, будет производиться поиск тех пикселей, которые предположительно относятся к разметке футбольной площадки. В итоге, в результате предобработки входящего кадра, система должна получить бинарное изображение, где черные пиксели обозначают линии разметки, а белые - весь остальной фон.

Затем системе необходимо определить прямые, лежащие на этом изображении, проходящие через найденные точки, а главное совпадающие с линиями разметки. После того как прямые найдены, пользователь должен будет сопоставить каждой из них соответствующую линию разметки реальной площадки, чтобы система смогла понять какой участок поля находится в кадре. Эти соответствия позволят программе вычислить положение видеокамеры относительно площадки по найденным прямым и их соответствиям.

При поступлении следующих кадров система будет рассматривать не всё изображение, а только лишь окрестности уже найденных на предыдущих кадрах прямых. Только они будут фильтроваться, и в них будут искаться точки поля и сама разметка. Это должно намного ускорить процесс поиска. На основе найденных линий разметки и предыдущей позиции видеокамеры, системе необходимо вычислить текущее положение камеры.

Под вычислением положения камеры, понимается нахождение матрицы перехода от координат на реальной площадке к координатам видео изображения.

Обработка всех последующих кадров должна проходить быстро, и укладываться в 40 милисекунд на один кадр.

# Алгоритмы предобработки изображения

Поступающие изображения с цифровой видео камеры имеют формат RGB. При анализе изображений намного удобнее работать с форматом YUV, так как в нем компонента Y определяет интенсивность, а компоненты U, V определяют цветность. Это позволяет избежать многих проблем с тенями. Дело в том, что при падении тени на объект, значение Y серьёзно уменьшается (при выходе из тени - увеличивается), а значения U, V практически не изменяются. И таким образом легко понять изменился ли объект в данной точке или же просто на него упала тень [1].

Переход из RGB в YUV осуществляется по следующим формулам:

(1)

(2)

(3)

Для упрощения анализа, изображение необходимо отфильтровать, с целью удаления шумов и лишних объектов, и только лишь потом искать линии. Так как разметка располагаются на площадке, то после фильтрации системе нужно найти саму площадку, а затем проводить на ней поиск линий разметки.

# Фильтрация изображения

Фильтрация изображений является одной из самых фундаментальных операций компьютерного зрения, распознавания образов и обработки изображений. Фильтрацией изображения называется операция, имеющая своим результатом изображение того же размера, полученное из исходного по некоторым правилам. Обычно интенсивность каждого пикселя результирующего изображения обусловлена интенсивностями пикселей, расположенных в некоторой его окрестности в исходном изображении [2]. Правила, задающие фильтрацию (их называют фильтрами), могут быть самыми разнообразными.

Например, линейные фильтры. Они представляют собой семейство фильтров, имеющих очень простое математическое описание. Вместе с тем они позволяют добиться самых разнообразных эффектов. Применение линейного фильтра к изображению – это по сути свертка изображения по заданной функции.

Свертка двумерной функции f по функции g в непрерывном и дискретном случае вычисляется следующим образом:

; (4)

Где – ядро (kernel) свертки размером x.

К линейным фильтрам относятся сглаживающие фильтры, которые действуют на изображение аналогично мутному стеклу: изображение становится нерезким, размытым. Простейший “прямоугольный” сглаживающий фильтр радиуса r задается при помощи матрицы размера (2r + 1) × (2r + 1), все значения которой равны выражении (5) , а сумма по всем элементам матрицы равна, таким образом, единице.

; (5)

При фильтрации с данным ядром значение пикселя заменяется на усредненное значение пикселей в квадрате со стороной 2r+1 вокруг него. Характерной чертой этого фильтра, является то, что образом белой точки на черном фоне будет равномерно размытый серый квадрат.

Одним из возможных применений сглаживающих фильтров является шумоподавление, т.е. задача восстановления исходного изображения, к пикселям которого добавлен случайный шум. Шум меняется независимо от пикселя к пикселю и, при условии, что математическое ожидание значения шума равно нулю, шумы соседних пикселей будут компенсировать друг друга. Чем больше окно фильтрации, тем меньше будет усредненная интенсивность шума, однако при этом будет происходить и существенное размытие значащих деталей изображения. Естественным предположением об исходном незашумленном изображении будет схожесть значений интенсивности пикселей, находящихся рядом. Причем чем меньше расстояние между пикселями, тем больше вероятность их похожести. Это и отличает исходное незашумленное изображение от шумовой компоненты, для которой схожесть пикселей никак не зависит от расстояния между ними [3]. Исходя из вышесказанного можно предположить, что шумоподавление при помощи прямоугольного фильтра имеет существенный недостаток: пиксели, находящиеся на расстоянии r от обрабатываемого, оказывают на результат тот же эффект, что и соседние.

Более эффективное шумоподавление можно, таким образом, осуществить, если влияние пикселей друг на друга будет уменьшаться с расстоянием. Этим свойством обладает гауссовский фильтр:

; (6)

Где , а параметр задает степень размытия.

Гауссовская фильтрация также является сглаживающей. Однако, в отличие от прямоугольного фильтра, образом точки при гауссовой фильтрации будет симметричное размытое пятно, с убыванием яркости от середины к краям, что гораздо ближе к реальному размытию от расфокусированных линз. Гауссовская фильтрация более эффективна при шумоподавлении: влияние пикселей друг на друга при гауссовой фильтрации обратно пропорционально квадрату расстояния между ними. Коэффициент пропорциональности, а следовательно, и степень размытия, определяются параметром σ. Фильтр Гаусса является высокочастотным фильтром, а значит, его выгодно применять для повышения резкости изображения [1].

Таким образом, при получении из видеопотока самого первого кадра, система должна применить к нему фильтр Гаусса, для удаления лишних шумов, а так же для увеличения резкости самого изображения. Это значительно облегчит поиск линий разметки, который будет основываться на резком изменении интенсивности точек на изображении. На рисунке 3 изображено входящее изображение, а на рисунке 4 то же изображение после применения к нему фильтра Гаусса.

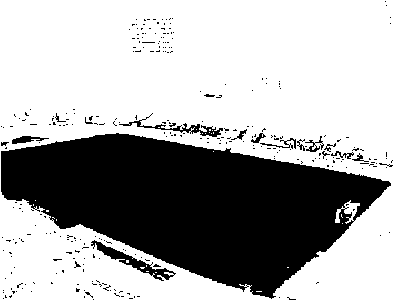
 

**Рис. 3. Рис. 4.**

# Построение бинарного изображения маски поля

При получении отфильтрованного изображения программе необходимо найти на нем точки, относящиеся к площадке. То есть система должна создать маску поля - бинарное изображение, черные точки которого будут обозначать поле. Для этого сначала нужно настроить систему. А именно ввести интервалы для компонентов U и V, по которым будут искаться точки, принадлежащие к футбольному полю. Вводить их должен пользователь. Для того чтобы настройка компонент U и V, проходила быстрее система должна выполнить программу по следующему алгоритму:

1. Разбить пространство (U, V) на ячейки размером 26x26. Так как U и V, то таких ячеек будет 100;
2. Найти ячейку, в которую попадает наибольшее количество пикселей изображения;
3. Вывести бинарное изображение с черными точками, координаты которых совпадают с координатами точек в данной ячейке;
4. Предоставить пользователю возможность: или подтвердить выбор данной ячейки или переключиться на следующую ячейку;
5. Если найденные точки не принадлежат точкам поля (рис. 5), то пользователь должен переключиться на следующую максимальную ячейку, а система в таком случае перейти на шаг 3. Если же найденные точки являются точками площадки (рис. 6), то система отмечает текущую ячейку, как искомую.



**Рис. 5. Рис. 6.**

После того как искомая ячейка найдена, системе необходимо вычислить среднее значение U, V в этой ячейке. После этого пользователь должен настроить радиус интервалов для U и V в отдельности. В дальнейшем все пиксели, попадающие в этот интервал, будут считаться принадлежащими к полю. Естественно, оба эти интервала в течение трансляции будут пересчитываться, но система будет это делать уже автоматически, находя математическое ожидание компонент U, V и их дисперсию после обработки каждого кадра.

Как видно на рисунке 6, не все точки, попавшие в интервал, принадлежат полю, поэтому необходимо отделить все ненужные области не относящиеся к футбольной площадке. Также бывают случаи, когда на самой площадке образуются мелкие “дыры”, которые нужно закрыть, ведь их расположение может совпасть с местами разметки поля, и в таком случае система не обнаружит эти линии разметки.

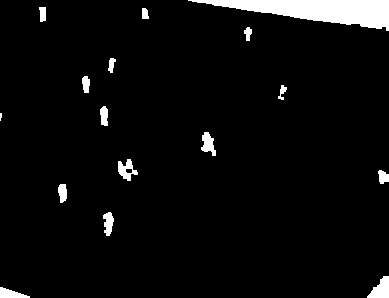
Для устранения “дыр” можно воспользоваться медианной фильтрацией, которая относиться к нелинейным фильтрам и определяется следующим образом:

(7)

То есть результат фильтрации есть медианное значение пикселей окрестности. Так как в бинарном изображении значение пикселей равно либо нулю, либо единице, то медианная фильтрация равносильна операции “Усреднение” (Значение пикселя будет равно тому, которое чаще встречается в окрестности этого пикселя).

В данном случае фильтр Гаусса не подходит, так как изображение бинарное. Вообще на цветных изображениях, шум с нулевым математическим ожиданием, добавленный к исходному сигналу, является только лишь одним из видов помех. Медианная фильтрация способна эффективно справляться с помехами в более общем случае, когда помехи независимо воздействуют на отдельные пиксели [2]. Преимущество медианной фильтрации перед линейной сглаживающей фильтрацией заключается в том, что яркий пиксель на темном фоне будет заменен на темный, а не "размазан" по окрестности.

На рисунках 7 и 8 показано применение к бинарному изображению медианной фильтрации с радиусом единица. Видно, что на рисунке 7 присутствуют белые пиксели, которые совпадают с линиями разметки, а это недопустимо. После применения медианной фильтрации такие пиксели перекрашиваются в черные, а белые пиксели соответствующие игрокам не изменяются (рис. 8).



**Рис. 7. Рис. 8.**

Для устранения крупных областей, не относящихся к полю, но попадающих в заданные интервалы U и V, можно воспользоваться рекурисвным алгоритмом выявления областей связанности. Связная область – это такое множество пикселей, у каждого из которых есть хотя бы один сосед, принадлежащий данному множеству [4].

Соседи пикселей:

4-связность 8-связность

Программа будет искать 4-связные области максимальных размеров. Код этой программы представлен в листинге 1. Функция *Segmentation*, получает на входе указатель на бинарное изображение (массив типа *bool*, размеров *height\*width*), а в результате работы отсекает все мелкие области, не относящиеся к полю.

int Limit=9500;

void Segmentation(bool \*mask)

{

bool \*mask2 = new bool[height\*width];

CopyMemory(mask2, mask, height\*width);

FillMemory(mask, height\*width, 1);

for (int x=0;x<width;x++)

for (int y=0;y<height;y++)

Fill(x,y,0, mask2);

delete mask2;

}

bool Fill(int x, int y, int L, bool \*mask2)

{

if (mask2[x+y\*width]==true)

return false;

mask2[x+y\*width]=true;

bool b=false;

if (L>Limit) b=true;

if (!b && y>0) b=Fill(x,y-1,L+1,mask2);

if (!b && x>0) b=Fill(x-1,y,L+1,mask2);

if (!b && y<height-1) b=Fill(x,y+1,L+1,mask2);

if (!b && x<width-1) b=Fill(x+1,y,L+1,mask2);

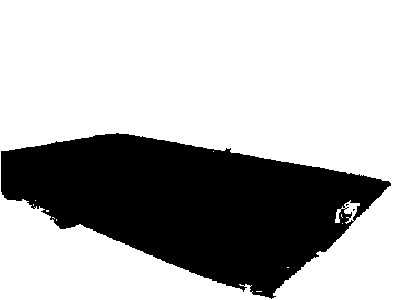
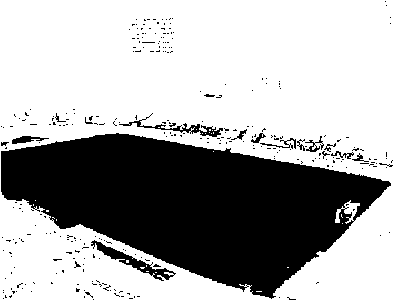
if (b) mask[x+y\*width]=false;

return b;

}

**Листинг 1.**

На рисунках 9 и 10 показаны бинарные изображения, соответственно, до и после применения рекурсивного алгоритма выявления максимальной области связанности.



**Рис. 9. Рис. 10**

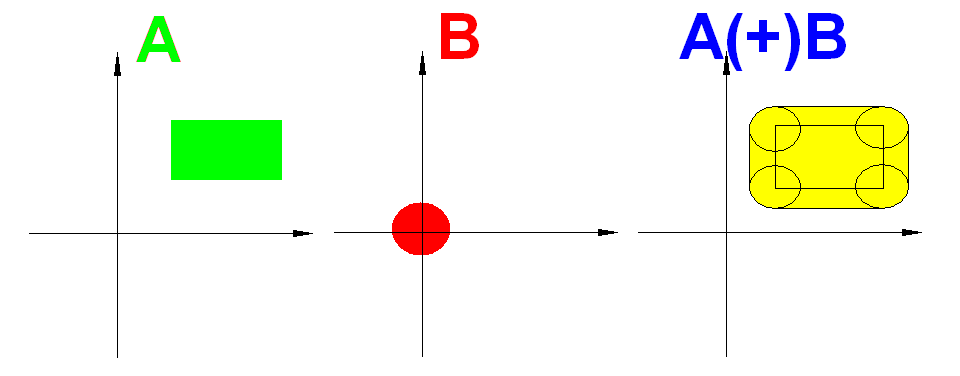
Также широко известный способ устранения шума в бинарных изображениях – это использование операций математической морфологии [5]:

1. Сужение (Erosion);
2. Расширение (Dilation);
3. Закрытие (Closing);
4. Раскрытие (Opening).

*Расширение*:

; (8)

где A,B - множества из (рис. 11).



**Рис. 11.**

*Сужение*:

; (9)

где – дополнение .

*Раскрытие*:

. (10)

*Закрытие*:

. (11)

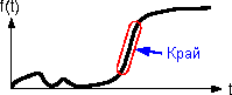
На полученном бинарном изображении для удаления лишних мелких объектов, не относящихся к площадке, можно воспользоваться операцией “Раскрытие” с большим радиусом множества . А последующее применение операции “Закрытие” устранит мелкие “дыры” на выделенной области.

Как показала практика, последовательное применение к бинарному изображению медианного фильтра и рекурсивного алгоритма выявления максимальной области связанности выполняется системой намного быстрее, чем последовательное применения операций “Раскрытие” и “Закрытие”, и при этом не теряется качество обработки.

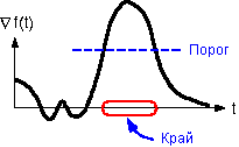
# Поиск разметки игровой площадки на изображении

На данном этапе система имеет два изображения: исходное изображение из видеопотока, “пропущенное” через фильтр Гаусса, а также бинарное изображение, которое определяет маску поля. Далее необходимо найти на площадке точки, относящиеся к разметке поля. Поиск разметки будет проходить на исходном, цветном изображении, прошедшем фильтр Гаусса. Но только лишь в тех точках, которые принадлежат маске поля, то есть лежат на площадке.

Для решения этой задачи можно воспользоваться различными известными алгоритмами поиска краев на изображении. Именно этот способ будет применен в системе. Для того чтобы понять откуда берутся края на изображении, необходимо понять что такое край. Край – это резкий переход яркости. Нахождение таких областей резкого изменения яркости можно организовать на основе анализа первой и второй производной изображения [6].

График функции: 

**Рис. 12.**

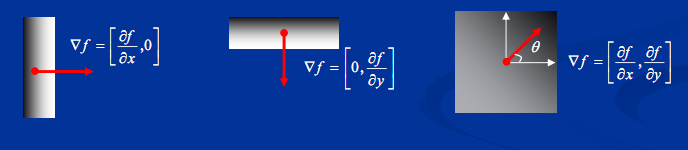
График производной: 

**Рис. 13.**

График 2-ой производной: 

**Рис. 14.**

Известно, что наибольшее изменение функции происходит в направлении её градиента. На рисунке 15 приведены примеры изменения яркости изображения и соответствующие им направления градиента. Величина изменения функции измеряется абсолютной величиной градиента.



**Рис. 15.**

Направление градиента вычисляется по формуле 12, его абсолютная величина по формуле 13, а угол между направлением градиента и осью 0X по формуле 14.

; (12)

; (13)

; (14)

“Направления края” задается перпендикулярным градиенту.

“Сила края” задается абсолютной величиной градиента (15).

; (15)

Приближенное вычисления градиента, можно вычислить по формуле (16).

(16)

Семейство методов поиска краев на изображении основано на приближенном вычислении градиента, анализе его направления и абсолютной величины. Для приближенного вычисления производной по направлению к изображению применяется чаще всего один из трех фильтров [6]:

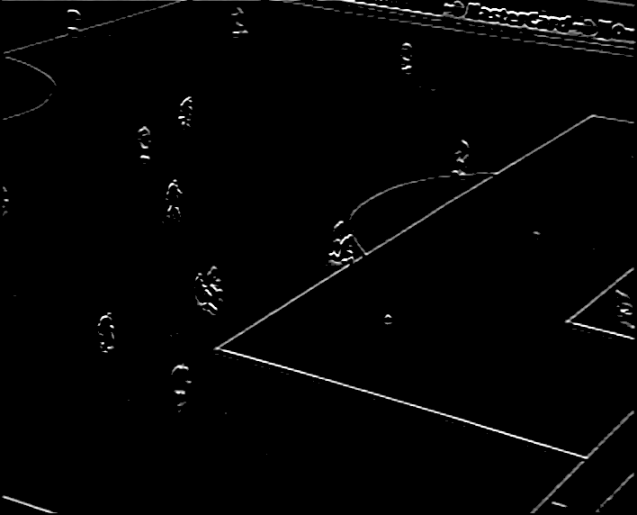
Фильтр Робертса

Фильтр Превитта

Фильтр Собеля

Практика показала, что наилучшим фильтром для решения данной конкретной задачи является фильтр Собеля, несмотря на то, что он медленнее фильтра Робертса, он более точно находит края на изображении.

На рисунке 16 показано исходное изображение, до этого прошедшее только через фильтр Гаусса, а на рисунке 17 показано это же изображение после применения к нему фильтра Собеля.

**Рис. 16. Рис. 17.**

Далее, к каждой точке изображения применяется заранее заданный порог по найденным абсолютным значениям градиента этой точки. Таким образом, система получит бинарное изображение с наиболее вероятными точками, принадлежащими к линиям разметки. Где белые точки - это те точки, у которых абсолютное значение градиента выше либо равно порогу, а черные – те, у которых ниже порога.

В итоге, проведя предварительную обработку изображения, последовательно выполняя следующие операции:

* изменение формата с RGB на YUV,
* применение к изображению фильтра Гаусса,
* поиск точек лежащих в заданном интервале для U и V,
* применение медианного фильтра к бинарному изображению,
* поиск максимальной области связанности,
* поиск краев с помощью фильтра Собеля в точках принадлежащих полю,
* применение к полученным значениям градиента каждой точки изображения, заранее заданного порога,

на выходе система получила бинарное изображение, с обозначенными предполагаемыми точками линии разметки.

# Алгоритмы поиска прямых на бинарном изображении.

После предобработки исходного изображения из видеопотока система получила бинарное изображение, в котором белыми точками обозначены предполагаемые линии разметки и возможно некоторый шум, а черными - остальной фон кадра. Теперь для того чтобы найти положение разметки необходимо найти прямые проходящие через эту разметку. При этом следует учитывать, что некоторые белые точки на полученном бинарном изображении могут не принадлежать линиям разметки поля. Для решения поставленной задачи нужно рассмотреть все возможные прямые, проходящие через изображение, после чего посчитать для каждой, сколько точек через нее проходит, и те прямые через которые проходит достаточное число точек считать существующими. Для того чтобы это сделать эффективно существует два подхода: преобразование Хафа и преобразование Радона [7].

Преобразование Радона пришло из томографии и изначально формулируется непрерывно, в то время как преобразование Хафа придумано специально для поиска фигур на изображениях, а следовательно дискретно изначально.

# Преобразование Хафа

Суть преобразования Хафа заключается в том, чтобы перебрать все белые точки на изображении, затем для каждой точки выбрать все прямые через нее проходящие, и увеличить счетчик точек для каждой из этих прямых.

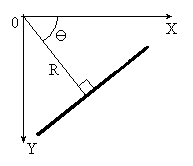
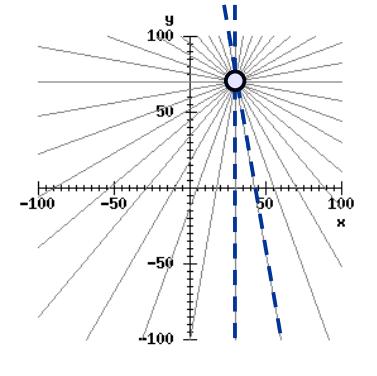
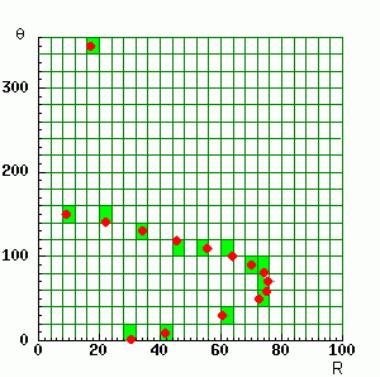
Прямую на плоскости можно задать следующим образом:

x\*cosθ+y\*sinθ=R (17)

где R – длина перпендикуляра, опущенного на прямую из начала координат, θ – угол между перпендикуляром к прямой и осью 0X.

Пусть начало координат изображения будет находиться в его левом верхнем углу, ось 0X лежит горизонтально и направлена слева направо, а ось 0Y вертикально сверху вниз (рис. 18), тогда угол θ изменяется в пределах (–π/2; π), а R положительно, и ограничено размерами изображения, то есть всегда меньше его диагонали.

Через одну точку на изображении можно провести несколько прямых, учитывая дискретность, их будет конечное число (рис. 19).

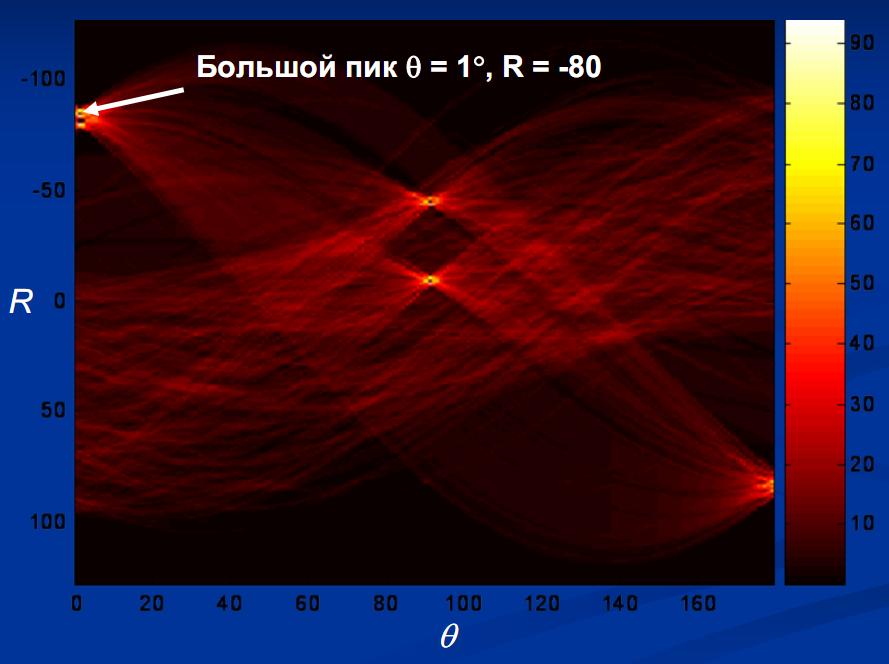
  

**Рис. 18. Рис. 19. Рис. 20.**

Каждой прямой пространства (x,y) соответствует точка фазового пространства (R,θ). Всем прямым проходящим через одну точку пространства (x,y) соответствует синусоида в пространстве (R,θ). Это показано на рисунке 19, где изображено пространство (x,y), и рисунке 20, где изображено фазовое пространство (R,θ).

Параметризация Хафа заключается в том, что в системе создается дискретное фазовое пространство (R, θ), которое получается из непрерывного, путем деления на ячейки. Каждой ячейке сетки [Ri x Ri+1] [θj x θj+1] ставится в соответствие счетчик. В дальнейшем, через каждую белую точку полученного бинарного изображения система будет проводить все возможные прямые, и для каждой прямой будет искать соответствующую ей точку в пространстве (R, θ). Счетчик той ячейки, в которую попадет точка, будет увеличиваться на единицу.

Когда система пройдет все точки изображения, и тем самым полностью заполнит фазовое пространство, ей будет необходимо найти положение наиболее вероятных прямых на исходном изображении, то есть прямых, у которых наибольший счетчик. На рисунке 21 показано заполненное фазовое пространство, где четыре скопления ярких точек соответствуют наиболее вероятным прямым.



**Рис. 21.**

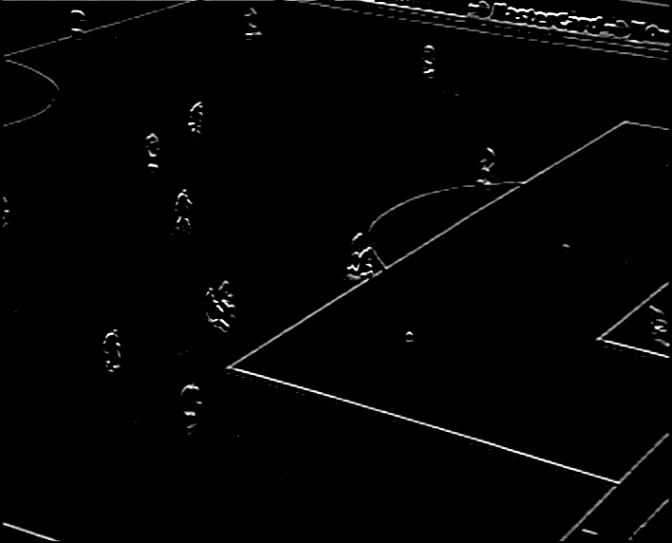
Если просто выбрать все те ячейки, в которых значение счетчика больше заранее заданного порога, то система найдет много прямых проходящих через почти одни и те же точки, то есть найдет лишние, “повторяющиеся” прямые. Для того чтобы избавиться от них, после заполнения фазового пространства система будет действовать по следующему алгоритму:

1. Поиск ячейки с максимальным значением счетчика. Если это значение меньше заранее заданного порога, то выход из алгоритма;
2. Прямую соответствующую найденной ячейки (R’,θ’) считать существующей;
3. Возвращение к исходному бинарному изображению и поиск белых точек, лежащих на найденной прямой. То есть таких точек, координаты которых (x,y), удовлетворяют уравнению (18).

x\*cosθ’+y\*sinθ’=R’; (18)

1. Каждую найденную точку перекрасить в черный цвет, и для каждой прямой проходящей через эту точку искать соответствующую точку в фазовом пространстве (R,θ). Счетчик той ячейки, в которую попадет точка, уменьшать на единицу;
2. Переход на шаг 1.

Таким образом, система найдет только необходимые прямые. На рисунке 22 показано бинарное изображение, а на рисунке 23 найденные прямые, наложенные на исходное изображение из видеопотока.

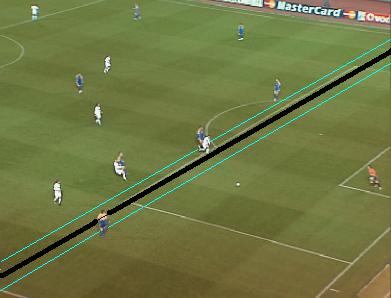
**Рис. 22. Рис. 23.**

# Итеративно перевзвешиваемые наименьшие квадраты.

Компьютерная программа на языке C++, в которой реализуется алгоритм поиска прямых на основе преобразования Хафа, выполняется в среднем за 250 милисекунд. Так как телевизионные камеры работают с частотой 50 кадров в секунду, то есть один кадр за 25 милисекунд, то такого рода программа просто будет не успевать за происходящими на изображении изменениями. А следовательно алгоритм необходимо оптимизировать.

Выполнять поиск прямых на основе преобразования Хафа, система будет всего один раз, в начале, на самом первом полученном изображении, для нахождения всех прямых расположенных на нем.

В новом поступающем изображении будет рассматриваться только некоторая окрестность прямых, найденных системой на предыдущем кадре (рис. 24).



**Рис. 24.**

В каждой такой окрестности система будет искать белые точки, и строить прямую, наилучшим образом их аппроксимирующую. Это означает прямую, на которой скорее всего лежат эти точки, т.е. прямую, для которой вероятность принадлежности этих точек максимальна (19).

; (19)

Прямая задается формулой (20).

; (20)

Пусть (Xi , Yi) – истинные точки, лежащие на искомой прямой, а

(xi , yi) –найденные белые точки, где .

Из-за шума (xi , yi) ≠ (Xi , Yi).

Вводиться предположение, что измерения порождены моделью и зашумлены нормально распределенным шумом (21,22):

; ; (21)

; ; (22)

Плотность распределения задается следующей формулой:

; (23)

Для решения задачи можно воспользоваться методом наименьших квадратов. Пусть нет ошибок в X, а есть только в Y. Все ошибки независимы, одинаково распределены и распределены нормально, то есть для любого выполняются равенства (24).

; ; ; (24)

Системе необходимо найти максимум правдоподобия (25). А так как ошибки независимы и нормально распределены, то из (25) выводиться (26).

; (25)

; (26)

Так как log – монотонная функция, то (26) равно (27).

; (27)

А так как ошибки одинаково распределены, то из (27) следует (28).

; (28)

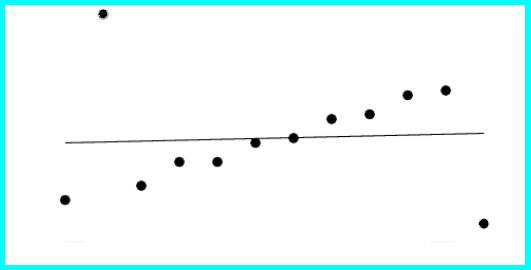
Уравнение (28) решается аналитически [8]. Находятся частные производные функции по *a* и по *b*. Затем они приравниваются к нулю и вычисляются неизвестные *a* и *b*.

В итоге, получаются формулы нахождения *a* и *b*:(29) и (30) соответственно:

; (29)

; (30)

Бывают ситуации, когда предположение о характере шума нарушается, то есть часть данных (полученных точек), не порождена искомой прямой. Тогда при оценке методом максимального правдоподобия результат может быть сколь угодно далек от истинного (рис. 25).



**Рис. 25.**

Очевидно, что системе необходимо уменьшить влияние “далеких” точек. Для этого можно модифицировать целевую функцию.

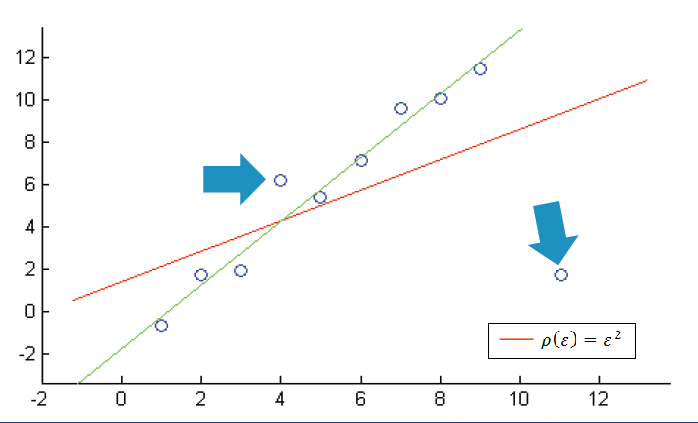
; (31)

В формуле (31) – целевая функция. Если , то учитывая (24) получается метод наименьших квадратов (28).

Необходимо выбрать функцию так, чтобы вклад далеких точек рос медленнее, например:

; (32)

При таком задании функции, значение при увеличении изменяется медленней, а следовательно, “далекие” точки будут незначительно влиять на положение прямой (рис. 26).



**Рис. 26.**

Часто используемые варианты функции [7]:

* В форме Тьюки: ; (33)
* В форме Коши: ; (34)

Настроечные константы: K=4.685; c=2.385;

Весьма не простая задача найти минимум указанных целевых функций . А в данном случае очень важна скорость выполнения программы. Поэтому для нахождения минимума лучше воспользоваться не самым точным, но довольно быстрым алгоритмом: методом последовательных приближений – итеративно перевзвешиваемых наименьших квадратов.

Рассмотрим взвешенные наименьшие квадраты:

; (35)

=1; (36)

Здесь - это “веса точек”.

Алгоритм итеративно перевзвешиваемых наименьших квадратов:

1. Получить начальное приближение методом наименьших квадратов . Номер итерации t=0.
2. Увеличить номер итерации.
3. Для посчитать текущую оценку шума .
4. Рассчитать веса точек с учетом функции .
5. Используя взвешенные наименьшие квадраты получить .
6. Если вектор параметров существенно изменяется (36), то перейти на шаг 2.

; (36)

К данному алгоритму необходимо добавить пояснения. Веса точек связанны с функцией следующим образом [7]:

* Тьюки: w; (37)
* Коши : ; (38)

А для того, чтобы рассчитать уровень шума в шаге 3, нужна устойчивая к выбросам оценка: , где – это несмещенная (для нормального распределения) робастная оценка средней ошибки. .

Этот метод позволяет довольно быстро и точно найти прямую, наилучшим образом аппроксимирующую белые пиксели в окрестности линий разметки предыдущего кадра.

В литературе основным недостатком данного метода указывается: необходимость хорошего первого приближения [7,8]. Но так как приближение у нас найдено на предыдущем шаге, это метод становиться очень удобным в применении к данной задаче.

# Алгоритм вычисления положения камеры по найденным прямым

Нахождение прямых это не окончательное решение основной задачи работы. На данном этапе необходимо узнать правило, по которому соотносятся точки изображения и точки реального пространства.

Вначале необходимо понять как получается изображение на кадре (рис 27).

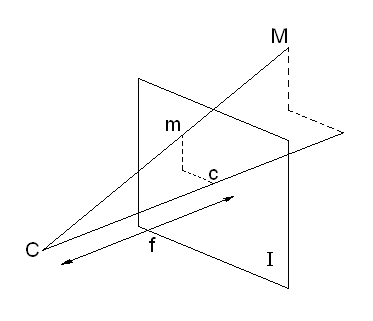
Изображение в видеокадре - это отображение сцены на плоскости.



**Рис. 27.**

# Принцип формирования изображения в видеокамере

Рассмотрим модель простейшей камеры-обскуры. Она удобна тем, что полностью описывается центром проекции и положением плоскости изображения. Поэтому проекция любой точки сцены на изображении может быть найдена как пересечение луча, соединяющего центр проекции и точку сцены, с плоскостью изображения (рис. 28).



**Рис. 28.**

Рассмотрим простейший случай, когда центр проекции камеры (фокус) помещен в начало системы координат, и плоскость изображения совпадает с плоскостью Z=1. Пусть (X,Y,Z) - координаты точки в 3-х мерном пространстве, а (x,y) - проекция этой точки на плоскость изображения. Перспективная проекция в этом случае описывается уравнениями (39), (40).

; (39) ; (40)

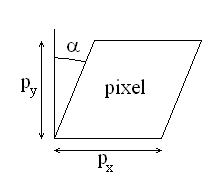
В матричной форме с использованием однородных координат эти уравнения переписываются в следующем виде:

(41)

Плоскость, расположенная на расстоянии 1 от центра проекции, и перпендикулярная оптической оси называется идеальной плоскостью изображения. Оптическая ось пересекает идеальную плоскость изображения в точке *с*, называемой принципиальной точкой.

Но такой простейший случай перспективной проекции практически всегда не соответствует реальной камере [9]. Расстояние от центра проекции до плоскости изображения, т.е. фокусное расстояние, обозначаемое *f*, обычно не равно 1. Также координаты точки в плоскости изображения могут не совпадать с абсолютными координатами. При использовании цифровой камеры, соотношение между координатами точки в изображении и абсолютными координатами точки на идеальной плоскости, определяется формой и размерами пикселей матрицы данной видеокамеры.

Обозначим принципиальную точку за [,, 1], размеры пикселя матрицы цифровой камеры за ,, а угол наклона пикселя за α (рис. 29).



**Рис. 29.**

Тогда координаты точки (x,y) в изображении, соответствующей точке на идеальной плоскости, определяются выражением:

; (42)

Матрица K называется матрицей внутренней калибровки камеры. В большинстве случаев у реальных цифровых камер угол наклона пикселей близок к прямому, т.е. , а ширина и высота пикселя равны [10]. Принципиальная точка обычно располагается в центре изображения. Поэтому матрица K может быть записана в виде:

; (43)

Теперь пусть M - точка сцены в 3-х мерном пространстве. Любое движение является евклидовым преобразованием пространства, поэтому в однородных координатах оно выражается так:

; (44)

где *R* - матрица вращения, *t*- вектор переноса.

Движение камеры относительно сцены эквивалентно обратному движению точек сцены относительно камеры, поэтому оно равно выражению (45).

; (45)

где *R*, *t* - матрица вращения и вектор перемещения камеры относительно сцены. Матрица *C* из (45) называется матрицей *внешней калибровки* камеры.

Таким образом, матрица *C* переводит координаты точек сцены из системы координат сцены в систему координат, связанную с камерой.

Из вышеизложенных выражений (41), (43) и (45) можно вывести выражение произвольной перспективной проекции для любой камеры, с произвольной ориентацией и положением в пространстве:

; (46)

В более краткой форме с учетом предыдущих обозначений эта формула может быть записана как:

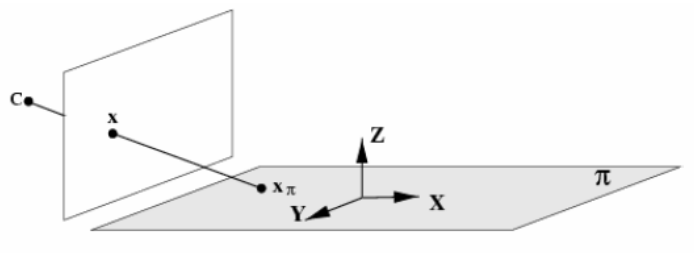
; (47)

Матрица называется матрицей проекции камеры.

# Вычисление матрицы гомографии

По аналогии с общим перспективным преобразованием можно рассмотреть перспективное преобразование плоскости.

Пусть плоскость *𝝅* совпадает с плоскостью (рис 30), то есть это плоскость земли, совпадающая с нашей спортивной площадкой.



**Рис.30.**

Тогда однородные трехмерные координаты любой ее точки M равны . Для любой камеры с матрицей проекции P, перспективное преобразование плоскости описывается матрицей размерности 3\*3. Такое преобразование плоскости называется гомографией [10]. В матричной форме гомография записывается как:

; (48)

Где *H* – матрица гомографии, *M* –точка на площадке, а *m* –точка, являющаяся проекцией *M* на плоскость изображения камеры.

Так как размеры площадки и положение её разметки известны, следовательно, известны и координаты (X,Y) любой конкретной точки поля. Так как прямые на исходном изображении уже найдены, значит, можно найти точки пересечения этих прямых. Учитывая, что при поступлении кадра, пользователь установил соответствия между реальной разметкой поля и найденными линиями на изображении, то система может сопоставить каждой найденной точке пересечения прямых, реальную точку пересечения линий разметки, координаты которых известны.

Пусть точка (x,y) лежит в плоскости земли на площадке, а точка (u,v) – это соответствующая ей проекция на изображение. Тогда

; (49)

Из (49) можно выразить координаты *u* и *v*:

; (50)

; (51)

Из выражений (50) и (51) можно построить два уравнения (52, 53):

; (52)

; (53)

Таким образом, зная одно соответствие (u,v)↔(x,y), можно построить два уравнения на параметры матрицы H.

Из следующего равенств (54) видно, что матрица H определяется с точность до множителя.

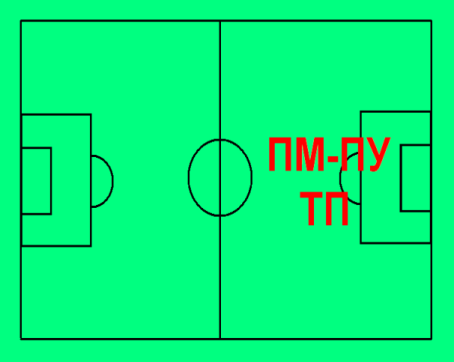
; (54)

Следовательно, один из параметров можно установить в 1. Пусть . Тогда получается 8 неизвестных параметров матрицы H. Как уже установлено: из одного соответствия точек получаем два уравнения, а значит для определения всей матрицы H необходимо и достаточно 4-х соответствий (u,v)↔(x,y). На футбольном поле, если взять любые две линии разметки, то они окажутся, либо параллельными, либо перпендикулярными. Значит для получения 4-х точек нам необходимо четыре прямые, две из которых параллельны между собой, а другие две перпендикулярны первым двум и также параллельны между собой.

Но не всегда в кадр попадает четыре прямые одновременно. Нужно постараться уменьшить минимальное количество необходимых прямых. Рассмотрим как задается положение видеокамеры в пространстве. Оно определяется тремя координатами (X,Y,Z) и тремя углами (α,β,γ) вокруг осей 0X, 0Y, 0Z соответственно. Камеры, берущие крупные планы, обычно стоят на одной начальной точке, и только лишь вращаются. Если мы имеем дело с такой камерой, то её координаты (X,Y,Z) не меняются, и угол вращения вокруг оси, перпендикулярной объективу камеры, тоже постоянен, так как камеры могут вращаться только в двух направлениях. Следовательно, изменяются только углы (α,β). Но у камер есть еще эффект Zoom. Он влияет на приближение, и отдаление, то есть на фокус *f*. В итоге получается три неизвестных (α,β,f), для вычисления которых требуется минимум три уравнения, а значит два соответствия (u,v)↔(x,y), или три прямые.

Итак, на первом кадре должны лежать минимум 4 прямых для вычисления гомографии. А на всех последующих достаточно будет трех прямых. После того как, система нашла три прямые и вычислила матрицу гомографии, ей необходимо, на основе этой матрицы, вычислить положение остальных линий разметки на кадре. Это делается для того, чтобы система, в случае когда одна из трех найденных прямых уйдет за кадр, начинала поиск новой линии разметки в окрестности одной из уже построенных прямых.

Матрица гомографии предназначена для того чтобы зная координаты площадки, вычислять соответствующие координаты на изображении кадра. Таким образом, системе остается задать изображение, которое необходимо наложить на поле, и координаты. На рисунках 31, 32 и 33 показано наложение виртуальной графики на кадр с футбольной площадкой: на рисунке 31 изображен входящий кадр; на рисунке 32 показана графика, которую необходимо наложить, и позиция куда накладывать; а на рисунке 33 изображен полученный кадр с виртуальной графикой.

**Рис. 31. Рис. 32. Рис. 33.**

# Полученные результаты

В результате проделанной работы, была написана система определения положения видеокамеры относительно пространства. Программа реализована на языке C++. Система настраивается пользователем в течение минуты, а затем начинает в реальном времени обрабатывать входящие изображения из видеопотока. В результате после обработки каждого кадра система вычисляет положение камеры относительно спортивной площадки, и на основе этого способна накладывать виртуальную графику на видеоизображение и передавать его на выход.

В данной работе были представлены различные алгоритмы фильтрации изображения. Подробно рассмотрены линейные и нелинейные фильтры:

* Усреднение;
* Фильтр Гаусса.
* Медианный фильтр.

Описан рекурсивный алгоритм выявления максимальной области связанности.

Рассмотрены методы устранения шума в бинарных изображениях с помощью операций математической морфологии:

* Сужение;
* Расширение;
* Закрытие;
* Раскрытие.

Описаны методы поиска краев на изображении на основе фильтров Робертса, Превитта и Собеля. Последний выбран как основной в написанной системе.

Рассмотрен алгоритм поиска прямых на изображении на основе преобразования Хафа. В данной работе этот алгоритм был немного доработан, для того чтобы не учитывать прямые, относящиеся к уже найденной линии разметки.

Предоставлены методы максимального правдоподобия, для построения прямой, наилучшим образом аппроксимирующей некоторое данное множество точек.

Обоснован выбор основного алгоритма поиска прямых на основе итеративно перевзвешиваемых наименьших квадратов.

Рассмотрен алгоритм вычисления положения видеокамеры по найденным прямым. В работе объясняется, какое минимальное количество прямых должно быть найдено на изображении для вычисления матрицы гомографии. Подробно описано как вычисляется матрица гомографии, если известны координаты видеокамеры, её углы поворота и фокус.

# Заключение

В настоящий момент написанная система, находиться на этапе тестирования. В дальнейшем планируется усовершенствование системы, путем вычисления позиций игроков и мяча на площадке, и соответственного наложения графики, такой как офсайдные линии, расстояние до ворот при штрафных ударах, и многое другое. Для этого будет достаточно определить позицию игрока на кадре, а вычисление его координат на площадке буде производиться с помощью уже найденной матрицы преобразования, которая значительно облегчает дальнейшее развитие системы.

Рассмотренные в данной работе методы и алгоритмы, могут быть легко перенесены на различные виды спорта с учетом их особенностей, или использованы для построения видео студий на телеканалах.

# Литература и ссылки

[1]<http://courses.graphicon.ru/files/courses/vision/2009/cv_2009_02.pdf> “Общая схема алгоритмов распознования” Курсы лаборатории компьютерной графики при ВМК МГУ. Антон Конушин.

[2] <http://www.intuit.ru/department/graphics/rastrgraph/8/> “Фильтрация изображений” Алгоритмические основы растровой графики. информация **[-]** Д.В. Иванов, А.А. Хропов, Е.П. Кузьмин, А.С. Карпов, В.С. Лемпицкий

[3] Ярославский Л.П. Введение в цифровую обработку изображений. – М.: Сов. радио, 1979

[4] <http://www.swsys.ru/index.php?page=article&id=1607> “Исследование методов сегментации изображений”. Грибков И.В., Захаров А.В., Кольцов П.П., Котович Н.В., Кравченко А.А., Куцаев А.С., Осипов А.С.

[5] Розенфельд А. Распознавание и обработка изображений с помощью вычислительных машин. – М.: Мир, 1972.

[6]<http://courses.graphicon.ru/files/courses/vision/2009/cv_2009_03.pdf> “ Выделение краев. Сегментация изображения” Курсы лаборатории компьютерной графики при ВМК МГУ. Вадим Конушин.

[7]<http://courses.graphicon.ru/files/courses/vision/2009/cv_2009_05.pdf> “ Введение в компьютерное зрение. Оценка параметров моделей” Курсы лаборатории компьютерной графики при ВМК МГУ. Антон Конушин, Ольга Баринова, Вадим Конушин, Антон Якубенко, Александр Велижев, Алексей Чернявский.

[8] <http://solidbase.karelia.ru/edu/meth_calc/files/12.shtm> “Численные методы поиска минимума функции нескольких переменных”.

[9] Бейтс Р., Мак-Доннелл М. Восстановление и реконструкция изображений. – М.: Мир, 1989.

[10] <http://cgm.computergraphics.ru/content/view/141> “Геометрические свойства нескольких изображений” Сетевой Журнал “Компьютерная Графика и Мультимедиа”. Антон Конушин.