Санкт-Петербургский Государственный Университет

Математико-механический факультет

Кафедра системного программирования

Система отслеживания документов на письменном столе

Курсовая работа студента 445 группы

Кривоконь Дмитрия Сергеевича

Научный руководитель А. Т. Вахитов

Санкт-Петербург

2010

Оглавление

1. Введение

В связи с бурным развитием технологий видеонаблюдения и в последние 10 лет стали применяться информационные системы на основе анализа видео в различных прикладных отраслях.

Эта работа посвящена системе слежения за документами, лежащими на письменном столе, при помощи, закрепленной над столом видеокамеры. Такая система позволяет регистрировать документы, попадающие на рабочий стол, определять их текущее местонахождение, считывать текст документов. Важной практически значимой задачей является скрытое наблюдение и анализ текстов документов. При этом автоматического распознавания текста не требуется, достаточно лишь повысить его читаемость.

Ранее для решения данной задачи была разработана система с ручным управлением. Над столом крепилась камера на механическом приводе, при помощи которого оператор наводил систему на интересующий его документ. При данном подходе система получалась крайне дорогой и сложной в обслуживании. Механический привод требовал постоянной поддержки, и кроме того, при скрытом наблюдении возникала проблема шума издаваемого им.

Исходя из этих проблем, появилась необходимость создания системы автоматического отслеживания документов. При таком подходе камера следит за всем столом сразу, и оператору необходимо просто указать тот документ, за которым нужно производить наблюдение. В системах данного вида, сразу же возникает проблема качества получаемых наблюдений, т.к. очевидно, что при наблюдении за поверхностью всего стола, качество отдельно взятых изображений документов будет значительно хуже, чем в системе с ручным управлением. Поэтому требуется проводить автоматическое улучшение качества изображения на основе длительных наблюдений объектов.

Целью данной работы было создание такой системы и изучение существующих алгоритмов для каждой из представленных выше задач. В качестве методов автоматического отслеживания документов были выбраны цветовая фильтрация и разность с фоном [6], а автоматическое улучшении качества изображений производилось при помощи достаточно известного алгоритма Irani-Peleg [4].

2. Система слежения за документами

2.1 Архитектура системы

Система разделена на три основных модуля:

1. Модуль слежения за документами – обеспечивает автоматизированное отслеживание выбранных документов
2. Модуль выравнивания изображений документов – представляет изображений объекта в единой ориентации (необходимо для модуля улучшения качества)
3. Модуль улучшение качества и разрешения

Базовую работу системы можно представить следующей схемой:

Оператор выделяет некоторый документ, указывая его положение на текущем кадре. Затем координаты передаются в модуль по слежению за объектами. Определив, какой объект был выделен, данный модуль будет следить за ним на последующих кадрах, предоставляя информацию о положении объекта оператору. Как только накопляется

достаточное количество наблюдений объекта (данная величина регулируется оператором), эти наблюдения передаются в модуль выравнивания изображений. После того, как все изображения были приведены к единой ориентации, полученные результаты передаются на вход алгоритму улучшения качества, а его результаты уже выдаются оператору.

2.2 Выделение и отслеживание документов

2.2.1. Цветовая фильтрация

Базовым предположением о документах является их цветовая однородность, а именно, то, что они представляют собой некие яркие, белые объекты. На основе этого возможно построить метод, выделяющий интересующие нас области на изображении, путем цветовой фильтрации. Фильтр представляет собой проверку пикселей документа на два условия:

1. Яркость должна быть выше определенного уровня
2. Цветовые компоненты должны отличаться между собой менее чем на определенную величину

Далее будет представлено формальное определение фильтра. Рассмотрим цветное изображение I:

$$I(x,y) = \left(I^{(R)}(x,y), I^{(G)}(x,y), I^{(B)}(x,y)\right)$$

Где $I^{(R)}$, $I^{(G)}$, $I^{(B)}$ - цветовые компоненты красного, синего и зеленого цвета соответственно. Введем для каждого пикселя его функцию яркости:

$$B_I(x,y) = \frac{I^{(R)}(x,y) + I^{(G)}(x,y) + I^{(B)}(x,y)}{3}$$

Также рассмотрим функцию отхождения каждой цветовой компоненты пикселя от его яркости:

$$V_I(x,y) = \sum_{C \in \{R,G,B\}} \left|I^{(C)}(x,y) - B_I(x,y)\right|$$

Тогда, введя пороговые значения B_{TH} и V_{TH}, бинарная маска на основе цветового фильтра будет выглядеть следующим образом:

$$Mask_I(x,y) = \begin{cases} 1, & B_I(x,y) > B_{TH} \; and \; V_I(x,y) < V_{TH} \\ 0, & \text{иначе} \end{cases}$$

Данный метод не отличает статические объекты от динамических, поэтому во многих случаях его применения недостаточно для успешного отслеживания документов. Совместно с ним в данной работе применялся алгоритм разности с фоном, который специально предназначен для детектирования движущихся объектов.

2.2.2. Разность с фоном

Т.к. в поставленной задаче камера находится в статическом положении, то для детектирования движения и появления объектов можно использовать методы, основанные на разнице с фоном. В таких методах пиксели изображения каждого кадра разделяются на две категории: пиксели фона (background), пиксели движущихся объектов (foreground). Разделение производится на основе разницы текущего кадра с текущей моделью фона. Для моделирования фона существует огромное количество методов, начиная от статического фона, когда в качестве модели используется один кадр, до методов, основанных на представлении фона при помощи смеси нормальных распределений (обзор методов можно найти в [5]).

В данной работе использовалась достаточно простая модель фона, состоящая из среднего значения и дисперсии. Обновление текущей модели производится при помощи метода скользящего среднего, который, в сущности, представляет собой взвешенное усреднение на каждой итерации новых данных с существующей моделью. Далее представлено описание алгоритма.

Пусть (M_{k-1}, D_{k-1}) – модель фона в момент времени $k-1$. Здесь M_{k-1} – среднее значение фона, а D_{k-1} – его дисперсия. Тогда при получении нового кадра F_k, модель обновляется согласно следующим правилам:

$$M_k(x,y) = \alpha * M_{k-1}(x,y) + (1-\alpha) * F_k(x,y)$$

$$D_k(x,y) = \beta * D_{k-1}(x,y) + (1-\beta) * \|F_k(x,y) - M_{k-1}(x,y)\|^2$$

Где $\alpha, \beta \in [0,1]$ – константы, регулирующие скорость обучения среднего значения и дисперсии соответственно. Начальное состояние модели задается как:

$$M_0(x,y) = F_0(x,y)$$

$$D_0(x,y) = \sigma^2$$

Где σ^2 - некоторая фиксированная константа, а в качестве F_0 выбирается первый кадр, либо среднее значение по нескольким начальным кадрам.

На основе этого бинарная маска, отделяющая фон от объектов, строится следующим образом:

$$Mask_k(x,y) = \begin{cases} 1, & \|F_k(x,y) - M_{k-1}(x,y)\|^2 > \gamma * D_{k-1}(x,y) \\ 0, & \text{иначе} \end{cases}$$

В представленной модели есть существенный недостаток – при ее обновлении учитываются пиксели, которые принадлежат движущимся объектам, что может негативно повлиять на качество работы алгоритма. Это можно исправить, проводя обновления только по тем пикселям, которые были охарактеризованы, как принадлежащие фону:

$$M_k(x,y) = \begin{cases} \alpha * M_{k-1}(x,y) + (1-\alpha) * F_k(x,y), & Mask_k(x,y) = 0 \\ M_{k-1}(x,y), & Mask_k(x,y) = 1 \end{cases}$$

$$D_k(x,y) = \begin{cases} \beta * D_{k-1}(x,y) + (1-\beta) * \|F_k(x,y) - M_{k-1}(x,y)\|^2, & Mask_k(x,y) = 0 \\ D_{k-1}(x,y), & Mask_k(x,y) = 1 \end{cases}$$

Теперь пусть $Mask_k^{Color}$ – маска, полученная при помощи цветового фильтра, описанного в предыдущем разделе, и $Mask_k^{Bkg}$ – результат работы алгоритма разности с фоном, объединим эти данные следующим образом:

$$Mask_k(x,y) = Mask_k^{Color}(x,y) \,\&\, Mask_k^{Bkg}(\mathrm{x},\mathrm{y})$$

Эта маска учитывает, как и цветовую однородность документов, так и то, что они представляют собой движущиеся объекты. Далее остается сделать некоторую пост-обработку и выделить объекты.

2.2.3. Пост-обработка и выделение объектов.

Для фильтрации маски применяется операция растяжения (dilation) [8]. Ниже на Рис. 1, 2 и 3 представлено ее действие на примере одного кадра. Данный фильтр преобразует изображение следующим образом:

$$I_{output}\,(x,y) = \max_{(x_1,y_1)\in V_{x,y}} I_{input}\,(x_1,y_1)$$

Где $V_{x,y}$ – некоторая окрестность пикселя (x,y). Т.е. значение каждого пикселя устанавливается в максимум по некоторой его окрестности, что, очевидно, приводит к заполнению неоднородностей в маске Рис. 2.

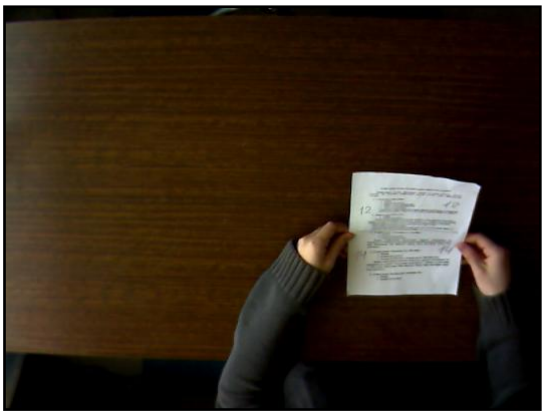

Рисунок 1 (оригинальный кадр)

Рисунок 2 (маска без фильтрации)

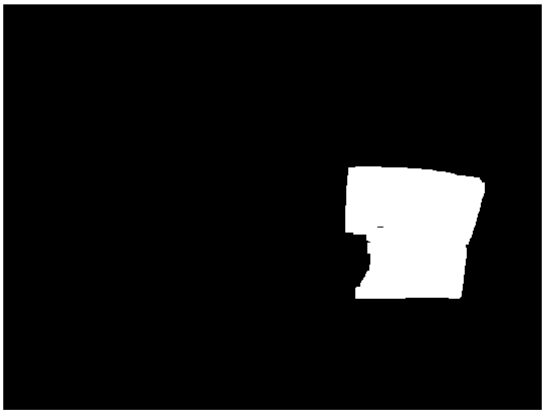

Рисунок 3 (маска после фильтрации)

После фильтрации маски выделение объектов происходит на основе Flood-Fill алгоритма, который выделяет связные компоненты белых областей на изображении. Связность определяется на основе пространственного расстояния между пикселями, точнее, два

пикселя считаются, принадлежащими одной компоненте связности, если расстояние между ними не превышает некоторой фиксированной величины. Выделив компоненты связности, мы получим список объектов на текущем кадре. Для того чтобы осуществлять слежение за каким либо объектом, нам необходимо сопоставлять его представление на предыдущем кадре с некоторым объектом текущего кадра. Это осуществляется путем минимизации расстояния между объектами:

$$O_{new} = \min_{O \in O_{cur}} d(O_{old}, O)$$

Где O_{new} - новое представление объекта, O_{cur} - список объектов на текущем кадре, O_{old} - представление объекта на предыдущем кадре. В качестве функции $d(\cdot,\cdot)$ можно использовать различные варианты расстояний. В данной работе применялась функция сравнения скорости и положения объектов:

$$d(O_1, O_2) = \alpha * \left\| V_{O_1} - V_{O_2} \right\| + \beta * \left\| P_{O_1} - P_{O_2} \right\|$$

Где V_O, P_O – скорость и положение объекта O соответственно, а α и β – некоторые весовые коэффициенты.

2.3 Приведение изображений документов к единой ориентации

В качестве результата работы предыдущих стадий системы получается набор неориентированных изображений объекта. Для работы алгоритмов повышения разрешения нам необходимо привести их к единой ориентации. Это осуществляется при помощи алгоритмов поиска образца.

Пусть $\{I_k\}_1^K$ – набор изображений, полученных в предыдущей стадии. В качестве образца ориентации возьмем первое из изображений – I_1. Задачей данной стадии работы системы заключается в нахождении преобразований $\{W_k\}_2^K$, каждое из которых переводит изображение I_k в I_1. Типы преобразований, вследствие постановки задачи, ограничиваются поворотами и сдвигами. Ниже приведено описание алгоритма Лукаса-Канаде и его модификации [6], которая применялась в данной работе.

2.3.1 Lucas-Kanade

Задачей данного алгоритма является поиск образца $T(v)$ в изображении $I(v)$, где $v = v(x, y)$ – вектор координат пикселя. Формально, необходимо найти такое преобразование $W: V_T \to V_I$, где V_T и V_I – это области координат изображений $T(v)$ и $I(v)$ соответственно, которое минимизирует сумму квадратов разностей:

$$\sum_{v \in V_T} \big(I\big(W(v)\big) - T(v)\big)^2$$

W ищется в некотором наборе параметризованных преобразований, т.е. $W = W(v, p)$. Принимая это во внимания, задача сводится к нахождению оптимального параметра p для функции:

$$F(p) = \sum_{v \in V_T} \big(I\big(W(v, p)\big) - T(v)\big)^2$$

Поиск p осуществляется при помощи итеративной процедуры:

$$p_k = p_{k-1} + \Delta p_{k-1} \quad (1)$$

$$\Delta p_{k-1} = H^{-1} \sum_{v \in V_T} \left[\nabla I \frac{\partial W}{\partial p}\right]^T (T(v) - I(W(v, p_{k-1})))$$

$$H = \sum_{v \in V_T} \left[\nabla I \frac{\partial W}{\partial p}\right]^T \left[\nabla I \frac{\partial W}{\partial p}\right]$$

Где ∇I – градиент изображения I, вычисляемый в точке $W(v, p_{k-1})$. Здесь гессиан H явно зависит от текущего приближения p_{k-1}, поэтому при каждой итерации приходится производить его пересчет и обращение, что, очевидно, весьма дорогостоящие операции. Данную проблему может решить Inverse-Compositional алгоритм.

2.3.2 Inverse-Compositional

В данном методе вместо аддитивной процедуры обновления (1) используется композиционный метод обновления, при котором происходит обновление не параметра, а самого преобразования, как функции от v:

$$W(v, p_k) = W(W(v, \Delta p_k), p_{k-1})$$

И на каждой итерации рассматривается следующая задача минимизации:

$$\sum_{v \in V_T} \left[T\big(W(v, \Delta p_k)\big) - I(W(v, p_{k-1}))\right]^2 \underset{\Delta p_k}{\longrightarrow} min$$

Ее решение:

$$\Delta p_k = H^{-1} \sum_{v \in V_T} \left[\nabla T \frac{\partial W}{\partial p}\right]^T \big(I\big(W(v, p)\big) - T(v)\big)$$

$$H = \sum_{v \in V_T} \left[\nabla T \frac{\partial W}{\partial p}\right]^T \left[\nabla T \frac{\partial W}{\partial p}\right]$$

Здесь Гессиан H вычисляется при значении параметра $p = 0$, следовательно, он не меняется от итерации к итерации. Более того, он никак не меняется при изменении изображения $I(v)$, что позволяет вычислять его только при инициализации метода.

3. Повышение качества и разрешения изображений (Super-Resolution)

3.1 Постановка задачи и метод максимального правдоподобия

Решение данной задачи основано на модели наблюдений, которая представляет собой некое приближение процессов происходящих при генерировании изображения в видеокамерах. Реальные наблюдения объекта (сигнал, получаемый из видеокамеры) рассматриваются как искаженные варианты изображения в улучшенном качестве, которое нужно найти:

Рисунок 4

К фильтрам относятся следующие виды искажений:

1. Геометрические (сдвиги, повороты, проективные преобразования)
2. Размытие, возникающее из-за оптических эффектов
3. Аддитивные шумы

Первый тип искажений, вообще говоря, не влияет на качество картинки, и компенсация его эффектов будет производиться при помощи алгоритмов поиска образца, описанных в разделе 2.3. Основным типом искажений, влияющим на качество изображения, является размытие. Данный эффект возникает из-за рассеивания лучей проходящих через линзу оптического прибора. На Рис.5 схематично представлено то, как это происходит в глазе человека:

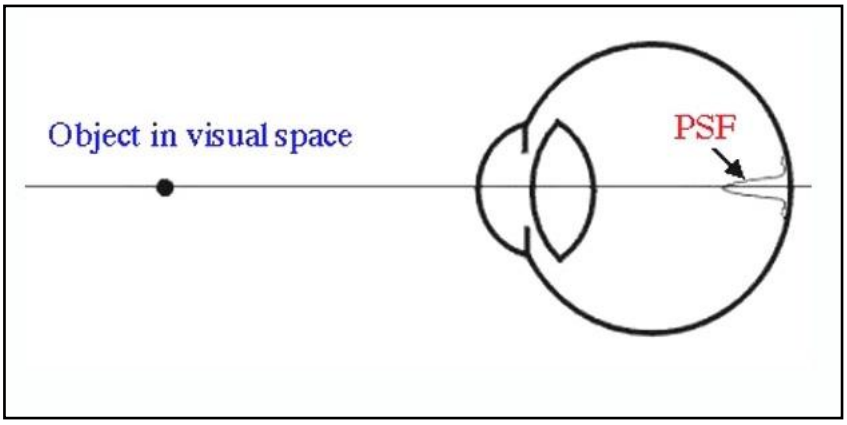

Рисунок 5 [2]

Изображения объекта, представляющего собой точку в реальном мире, на сетчатке глаза будет выглядеть как некоторая площадь, при этом уровень интенсивности воспринимаемого света, будет уменьшаться при увеличении расстояния от центра площади. Данный эффект можно моделировать при помощи специальных функций распределения (point-spread function, PSF), которые будут указывать своим значением уровень интенсивности.

Всю выше описанную модель можно записать формально в виде:

$$g_k = S \downarrow \left(h * T_k(f)\right) + \mu \quad (1)$$

Где g_k – наблюдаемые изображения, f – искомое изображение с улучшенным качеством, S – оператор понижения разрешения, h – point-spread function, T_k – оператор геометрического преобразования, μ – аддитивный шум. Умножение здесь представляет операцию дискретной свертки.

Основной задачей последующих алгоритмов будет подавление воздействия функции h. Эффекты аддитивного шума, в предположении его центрированности, подавляются за счет достаточного количества входных наблюдений и усреднения по ним. И как уже упоминалось, нахождение операторов T_k является отдельной задачей, решение которой было описано ранее.

Исходя из модели (1), для нахождения f применяется метод максимального правдоподобия ([1], [3]). Суть его заключается в максимизации следующей вероятности:

$$P(g_k|\hat{f}) = \prod_{\forall x,y} \frac{1}{\sigma\sqrt{2\pi}} \exp[\frac{-(\hat{g}_k(x,y) - g_k(x,y))^2}{2\sigma^2}]$$

Где $\hat{f}$ – некоторая оценка f, а $\hat{g}_k$ – приближение g_k, полученное из $\hat{f}$ по формуле (1) .

Далее рассмотрим задачу максимизации не самой вероятности $P(g_k|\hat{f})$, а ее логарифма:

$$L(g_k) = \log(P(g_k|\hat{f})) = -\sum_{\forall x,y} (\hat{g}_k(x,y) - g_k(x,y))^2$$

В итоге решение определяется следующим образом:

$$f = argmax \sum_k L(g_k) \quad (2)$$

Найти f можно, воспользовавшись матричным представлением модели наблюдений (1):

$$g_k = M_k f + \mu \quad (3)$$

$$g = \begin{bmatrix} g_1 \\ \dots \\ g_k \end{bmatrix} \quad M = \begin{bmatrix} M_1 \\ \dots \\ M_k \end{bmatrix}$$

$$g = Mf + \mu \quad (4)$$

Где каждая матрица M_k представляет собой действие всех операторов из (1), такое представление, очевидно, возможно (при условии линейности операторов T_k).

Теперь, подставляя (3) в (2) и принимая во внимание (4), мы можем перейти к следующей задаче минимизации:

$$\left\| M\hat{f} - g \right\|^2 \to \min \quad (5)$$

Задача (5) – стандартная задача линейных наименьших квадратов. Ее решение можно выписать в явном виде:

$$\hat{f} = (M^T M)^{-1} M^T g$$

Основной проблемой данного метода является вычисление матрицы $(M^T M)^{-1} M^T$. Матрица M представляет собой разреженную матрицу размера $(K * N_1) \times N_2$, где K - количество кадров, N_1 - количество пикселей в изображении низкого разрешения, N_2 - количество пикселей в изображении высокого разрешения. Очевидно, что вычисление ее псевдообратной матрицы $(M^T M)^{-1} M^T$ крайне трудоемко. Поэтому необходимо использовать различные итеративные методы. В данной работе в качестве такого метода использовался алгоритм Irani-Peleg [4], который будет описан ниже.

3.2. Алгоритм Irani-Peleg

Данный алгоритм представляет собой итеративную процедуру :

$$\hat{f}^{n+1} = \hat{f}^{n} + \frac{1}{K}\sum_{k=1}^{K} T_k^{-1}\left[\left(S \uparrow \left(g_k - \hat{g}_k^{(n)}\right)\right) * p\right] \quad (1)$$

Алгоритм сходен по своим принципам с алгоритмами градиентного спуска. На каждой итерации текущее приближение корректируется при помощи проектирования ошибок в область изображения в высоком разрешении. Проектирование ошибок осуществляется при помощи функции обратной проекции (back-projection function, BPF) p, которая аналогична по своему действию с PSF. Выбор функции p определяется из анализа сходимости алгоритма.

Для анализа авторы рассматривают несколько упрощенный вариант процедуры (1), а именно, они опускают оператор S, т.е. рассматривают задачу улучшения качества, без увеличения разрешения. На самом деле, это не умаляет общности, т.к. мы можем применить оператор S заранее к наблюдаемым изображениям объекта. При данном предположении верна следующая теорема:

Теорема. Итеративная процедура (1) сходится к желаемому результату, т.е. к такому f, что $\forall k\ g_k = T_k(f) * h,$ если выполнено следующее условие:

$$\|\delta - h * p\|_2 < \frac{1}{\frac{1}{K}\sum_{k=1}^{K}\|T_k\|_2} \quad (2)$$

$$\delta(x) = \begin{cases} 1, & x \in \left[-\frac{1}{2}, \frac{1}{2}\right] \\ 0, & \text{иначе} \end{cases}$$

Замечание. Если операторы T_k состоят только из 2D сдвигов и поворотов, то условие (2) преобразуется в:

$$\|\delta - h * p\|_2 < 1$$

При этом, чем меньше значение $\|\delta - h * p\|_2$, тем быстрее алгоритм (1) будет сходиться. Исходя из полученных условий, авторы советуют выбирать p следующим образом:

$$p = h^k, \qquad k = 1,2, \ldots$$

Основной проблемой данного метода является его медленная сходимость. Ниже на Рис 3. представлен его график сходимости в сравнении с графиком сходимости метода сопряженных градиентов примененного к задаче линейных наименьших квадратов описанной в предыдущем разделе:

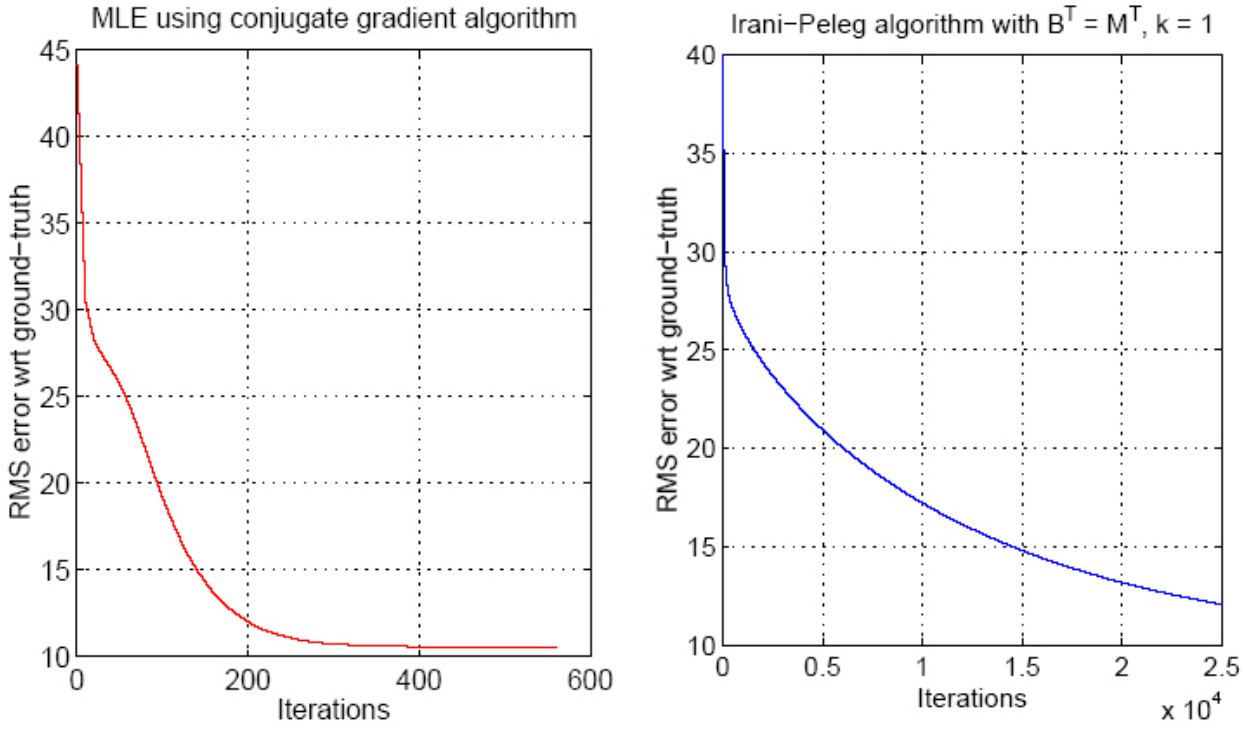

Рисунок 6 [1]

Большое количество итерации компенсируется простотой алгоритма и используемых в нем операций.

4. Результаты

Результатом этой работы является система слежения за документами на письменном столе, позволяющая просматривать документы в увеличенном разрешении. Были решены следующие конкретные задачи:

1. Осуществлена декомпозиция системы на модули и проект общей архитектуры системы
2. Для каждого модуля проведен обзор известных методов и выбраны методы, наиболее подходящие к специфике системы.
3. Проведена интеграция модулей.
4. Изучены основные подходы к решению задач повышения качества и разрешения.

Созданная система позволила добиться существенного увеличения читаемости текста на предварительных испытаниях.

Список Литературы

1. Super-resolution Enhancement of Text Image Sequences / Louka Dlagnekov // CSE 252C: Selected Topics in Vision & Learning, Department of Computer Science and Engineering University of California, San Diego

2. Visual Acuity / Michael Kalloniatis and Charles Luu // http://www.ncbi.nlm.nih.gov/bookshelf/br.fcgi?book=webvision&part=ch25kallspatial

3. Super-resolution Enhancement of Text Image Sequences / David Capel and Andrew Zisserman // Proceedings of the International Conference on Pattern Recognition (2000)

4. Motion analysis for image enhancement: resolution, occlusion, and transparency / M. Irani and S. Peleg // Journal of Visual Communication and Image Representation, 4:324–335, 1993.

5. Background subtraction techniques: a review / Massimo Piccardi // The ARC Centre of Excellence for Autonomous Systems (CAS) Faculty of Engineering, UTS, April 15, 2004

6. Lucas-Kanade 20 Years On: A Unifying Framework Part 1 / Simon Baker and Iain Matthews // International Journal of Computer Vision. 2004, Vol. 56, No. 3, P. 221-255

7. Limits on Super-Resolution and How to Break Them / Simon Baker and Takeo Kanade // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2002, Vol. 24, Issue 9, P. 1167-1183

8. Morphology Fundamentals: Dilation and Erosion // http://www.mathworks.com/access/helpdesk/help/toolbox/images/f18-12508.html