Правительство Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Санкт-Петербургский государственный университет»

Кафедра системного программирования

Болонкин Максим Викторович

Применение регистрации изображений к задаче повышения разрешения

Магистерская диссертация

Допущена к защите
Зав. кафедрой:
д. ф.-м. н., проф. Терехов А.Н.

Научный руководитель:
к. ф.-м. н., доц. Вахитов А.Т.

Рецензент:
Николаев А., аспирант

Санкт-Петербург
2015

Saint-Petersburg State University

Chair of Software Engineering

Bolonkin Maksim

Application of image registration to superresolution problem

Master's Thesis

Admitted for defence
Head of the chair:
Professor Andrey Terehov, Ph.D.

Scientific supervisor:
Associate professor Alexander Vakhitov, Ph.D.

Reviewer:
Alexander Nikolaev, Ph.D. candidate

Saint-Petersburg
2015

Оглавление

1 Введение **4**
1.1 Область исследований и ее актуальность 4
1.2 Алгоритм регистрации Лукаса–Канаде 7
1.2.1 Пирамидальная реализация алгоритма 12
1.3 Сегментация на основе Марковской сети 14

2 Обзор работы **17**
2.1 Постановка задачи 17
2.2 Применение маски в алгоритме регистрации 17
2.3 Алгоритм регистрации с адаптивной маской 22
2.4 Программная реализация 24
2.4.1 Описание классов библиотеки 26

3 Результаты экспериментов **29**
3.1 Эксперименты с синтетическими данными 29
3.2 Эксперименты с реальными данными 31

4 Заключение **35**

Глава 1

Введение

1.1 Область исследований и ее актуальность

Под разрешением изображения понимают меру детализации, то есть насколько точно передается зафиксированная на изображении сцена реального мира. При этом разрешение изображения можно рассматривать в нескольких различных аспектах. Пиксельное разрешение зависит от размеров пикселей камеры — чем меньше пиксель при фиксированных настройках оптической системы, тем более высокое разрешение будет иметь изображение. Пространственное разрешение определяется тем, насколько близкими могут быть две прямые линии, чтобы на изображении они выглядели отличными друг от друга. Это разрешение ограничивается оптическими свойствами фотографирующей системы и плотностью расположения пикселей на матрице в следствии дифракции света. Кроме того выделяют спектральное разрешение (насколько точно передаются цвета различных длин волн), радиометрическое разрешение (насколько точно передается яркость) и временное разрешение (насколько часто фиксируются кадры при съемке видео).

Таким образом, разрешение изображения определяется двумя основными факторами. Размытие в результате оптических свойств системы и посторонних процессов (например, физических свойств атмосферы или размытия из-за движения) понижает разрешение изображения, а низкая плотность сенсоров фотографирующей системы приводит к появлению эффекта алиасинга.

Повышение разрешения (superresolution) — это общий термин, который применяют для описания методов и техник улучшения качества изображений путем получения изображений более высокого разрешения. Большинство методов повышения разрешения основываются на исследовании двух и более изображений низкого разрешения, с целью получить изображение более высокого разрешения, которое содержало бы в себе всю информацию из изображений низкого разрешения. Если изображения низкого разрешения абсолютно идентичны, то никакой новой информации они добавить не могут и в этом случае повысить разрешение невозможно. Таким образом, все методы повышения разрешения основываются на идее, что камера или сцена никогда не бывают статичны, всегда происходят взаимные микродвижения, в результате которых последовательные изображения незначительно отличаются друг от друга, что позволяет строить на их основе изображение более высокого разрешения.

Методы повышения разрешения, как правило, состоят из трех или более этапов (Рис. 1.1). Первый — определение на суб-пиксельном уровне сдвига одного изображения относительно другого [14, 11, 10, 19, 8, 22]. Этот этап может быть выполнен с помощью алгоритмов регистрации изображений и нахождения оптического потока. Второй этап — соединение информации со всех изображений в одно изображение повышенного разрешения. Третий — интерполяция пикселей нового изображения, фильтрация и избавление от эффекта алиасинга. Так как зачастую сдвиги изображений измеряют-

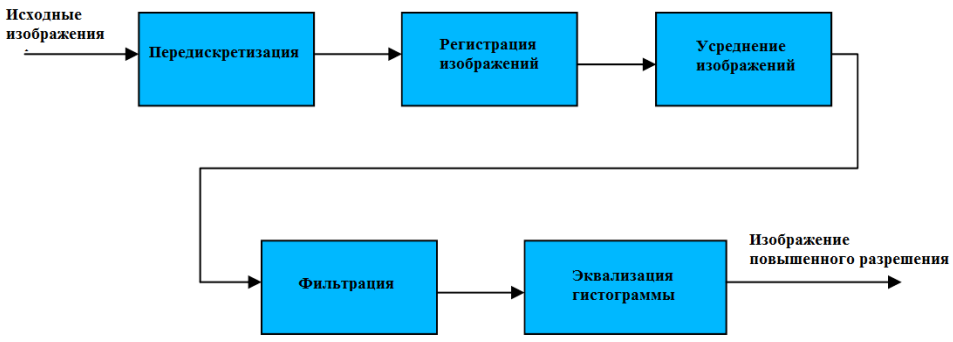


Рис. 1.1: Типичный алгоритм повышения разрешения

ся долями пикселя, крайне важно, чтобы первый этап давал как можно более точное значение сдвига, поэтому алгоритмам регистрации изображений посвящено значительное количество исследований, призванных улуч-

шить точность или ускорить процедуру регистрации. Один из самых первых алгоритмов был предложен Лукасом и Канаде [16], он представляет из себя итеративный градиентный спуск, минимизирующий квадратичную разность изображений. В дальнейшем было предложено несколько улучшений этого алгоритма, например, пирамидальная реализация [4]. Другие алгоритмы основываются либо на альтернативных метриках, например, на взаимной корреляции [1, 3] или взаимной информации [20, 17], либо на алтренативных методах оптимизации (минимизации квадратичной разности, максимизации взаимной информации) — например, метод MAP (максимальной апостериорной вероятности) [2], метод обратных проекций [23, 22] и другие [12, 24, 18, 7, 11].

Несмотря на то, что задача повышения разрешения имеет приложения практически во всех областях, связанных с видео и изображениями, наиболее разработанными являются методы регистрации изображений для медицинских целей (снимки МРТ, КТ и т.д.) [18, 7]. В сети Интернет доступно несколько библиотек для регистрации изображений с открытым исходным кодом, например, Image Registration Toolkit (ITK)[1], The Slicer Registration Case Library [2], NiftyReg [3]. Основные методы регистрации (например, Лукаса-Канаде) реализованы в этих библиотеках [9, 13]. Однако, при применении подобных библиотек к некоторым типам задач возникают сложности. Во-первых, все доступные библиотеки регистрации изображений спроектированы для использования в обработке медицинских изображений, отличительное особенностью которых является черный фон и светлое изображение. Значительное количество практических применений задачи повышения разрешения имеют в качестве фона либо светлые тона, либо неоднородное изображение, и применение к подобным изображениям методов из имеющихся библиотек дает плохие результаты. Другая сложность заключается в необходимости регистрации не всего изображения в целом, а какой-то его части (например, автомобильного номера вместо всей сцены с машиной) при перемещении этой части на фиксируемой сцене. Как следствие, интересующий регион на изображении сдвигается

[1] http://www.itk.org

[2] http://wiki.slicer.org/slicerWiki/index.php/Documentation/Nightly/Registration/RegistrationLibrary

[3] http://cmictig.cs.ucl.ac.uk/wiki/index.php/NiftyReg

при неподвижном фоне, что добавляет погрешности в применяемые методы регистрации.
Цель данной работы заключается в исследовании описанной проблемы и разработке методов регистрации, которые позволяли бы более точно определять взаимный сдвиг регистрируемых областей, что позволило бы получать изображения повышенного разрешения.

1.2 Алгоритм регистрации Лукаса–Канаде

Итеративный алгоритм регистрации изображений был предложен Лукасом и Канаде в 1981 году [16]. В дальнейшем было предложено несколько улучшений этого алгоритма в плане скорости сходимости [4].

Пусть $I(\mathbf{x})$ и $J(\mathbf{x})$ — функции двух переменных, $\mathbf{x} \in \mathbb{R}^2$, которые в целых значения координат дают значения пикселей двух изображений. Задачей регистрации изображений является нахождение такого преобразования координат $\mathcal{F} : \mathbb{R}^2 \to \mathbb{R}^2$, которое минимизирует некоторую функцию разности изображений в пределах некоторого интересующего региона $\mathfrak{R} \subset \mathbb{R}^2$. Самые распространенные функции разности между изображениями - это

- $L_1 = \sum\limits_{\mathbf{x}\in\mathfrak{R}} |I(\mathbf{x}) - J(\mathcal{F}(\mathbf{x}))|$
- $L_2 = \left(\sum\limits_{\mathbf{x}\in\mathfrak{R}} (I(\mathbf{x}) - J(\mathcal{F}(\mathbf{x})))^2 \right)^{\frac{1}{2}}$
- Отрицательная нормализованная корреляция

$$C = \frac{-\sum\limits_{\mathbf{x}\in\mathfrak{R}} I(\mathbf{x}) J(\mathcal{F}(\mathbf{x}))}{\left(\sum\limits_{\mathbf{x}\in\mathfrak{R}} I(\mathbf{x})^2 \right)^{\frac{1}{2}} \left(\sum\limits_{\mathbf{x}\in\mathfrak{R}} J(\mathcal{F}(\mathbf{x}))^2 \right)^{\frac{1}{2}}}$$

В качестве преобразования координат могут выступать различные геометрические преобразования, такие как сдвиг, поворот, растяжение, наклон, радиальная дисторсия и другие. Однако для наших целей мы ограничимся только аффинными преобразованиями, которые будем описывать матри-

цей A и вектором сдвига b. Таким образом,

$$\mathcal{F}(\mathbf{x}) = A\mathbf{x} + b, \quad A = \begin{pmatrix} 1 + v_{xx} & v_{xy} \\ v_{yx} & 1 + v_{yy} \end{pmatrix} \quad b = \begin{pmatrix} v_x & v_y \end{pmatrix}^T$$

В качестве функции разности принято брать норму L_2 [16, 4, 2]. Таким образом, задача регистрации изображений сводится к задаче минимизации функции

$$\epsilon(b, A) = \epsilon(v_x, v_y, v_{xx}, v_{xy}, v_{yx}, v_{yy}) = \sum_{\mathbf{x} \in \mathfrak{R}} (I(\mathbf{x}) - J(A\mathbf{x} + b))^2 \tag{1.1}$$

Обозначим через D градиент функции ошибки ϵ по параметрам v_x, v_y, v_{xx}, v_{xy}, v_{yx}, v_{yy}:

$$D = \begin{pmatrix} \frac{\partial \epsilon}{\partial v_x} & \frac{\partial \epsilon}{\partial v_y} & \frac{\partial \epsilon}{\partial v_{xx}} & \frac{\partial \epsilon}{\partial v_{xy}} & \frac{\partial \epsilon}{\partial v_{yx}} & \frac{\partial \epsilon}{\partial v_{yy}} \end{pmatrix}$$

В точке минимума градиент обращается в нулевой вектор.
Продифференцируем уравнение (1.1) по параметрам и получим:

$$D = -2 \sum_{\mathbf{x} \in \mathfrak{R}} (I(\mathbf{x}) - J(A\mathbf{x} + b)) D_2(\mathbf{x}) \tag{1.2}$$

где

$$D_2(x) = \begin{pmatrix} \frac{\partial J}{\partial x} & \frac{\partial J}{\partial y} & x\frac{\partial J}{\partial x} & y\frac{\partial J}{\partial x} & x\frac{\partial J}{\partial y} & y\frac{\partial J}{\partial y} \end{pmatrix} \tag{1.3}$$

Затем подставим вместо $J(Ax+b)$ линейную часть разложения этой функции в ряд Тейлора в окрестности точки 0 (так как предполагается небольшой сдвиг, данную замену можно рассматривать как хорошее приближение):

$$J(A\mathbf{x} + b) \approx J(\mathbf{x}) + D_2(x)\bar{v}$$

где $\bar{v} = \begin{pmatrix} v_x & v_y & v_{xx} & v_{xy} & v_{yx} & v_{yy} \end{pmatrix}^T$ - это расширенный вектор неизвестных параметров.

Подставляя разложение для J в уравнение (1.2), получим новое выражение

$$D \approx -2 \sum_{\mathbf{x} \in \mathfrak{R}} (I(\mathbf{x}) - J(x) - D_2(\mathbf{x})\bar{v})) D_2(\mathbf{x}) \tag{1.4}$$

Величина $I(x) - J(x)$ представляет собой простую разность между значе-

ниями пикселей изображений

$$\delta I(\mathbf{x}) = I(\mathbf{x}) - J(\mathbf{x})$$

Несмотря на то, что уравнение (1.3) определяет, что градиентный вектор $D_2(x)$ следует считать от функции $J(x)$, его вполне можно вычислять и от функции $I(x)$, так как взаимное смещение между изображениями предполагается достаточно маленьким. В итеративной реализации алгоритма эта замена дает вычислительное преимущество. В связи с этим введем новое обозначение

$$\nabla I = \begin{pmatrix} I_x \\ I_y \\ xI_x \\ yI_x \\ xI_y \\ yI_y \end{pmatrix} = D_2(x)^T$$

Для вычисления частных производных используется оператор центральной разности

$$I_x(\mathbf{x}) = \frac{\partial I(x)}{\partial x} = \frac{I(x+1,y) - I(x-1,y)}{2}, \tag{1.5}$$

$$I_y(\mathbf{x}) = \frac{\partial I(x)}{\partial y} = \frac{I(x,y+1) - I(x,y-1)}{2}. \tag{1.6}$$

С новыми обозначениями уравнение (1.4) можно переписать в виде

$$\frac{1}{2}D^T \approx \sum_{\mathbf{x}\in\mathfrak{R}}((\nabla I \nabla I^T)(v) - \bar{\nabla} I \delta I) \tag{1.7}$$

Введем обозначения

$$\mathcal{G} = \sum_{\mathbf{x}\in\mathfrak{R}} \nabla I \nabla I^T = \sum_{\mathbf{x}\in\mathfrak{R}} \begin{pmatrix} I_x^2 & I_xI_y & xI_x^2 & xI_x^2 & xI_xI_y & yI_xI_y \\ I_xI_y & I_y^2 & xI_xI_y & yI_xI_y & xI_y^2 & yI_y^2 \\ xI_x^2 & xI_xI_y & x^2I_x^2 & xyI_x^2 & x^2I_xI_y & xyI_xI_y \\ yI_x^2 & yI_xI_y & xyI_x^2 & y^2I_x^2 & xyI_xI_y & y^2I_xI_y \\ xI_xI_y & xI_y^2 & x^2I_xI_y & xyI_xI_y & x^2I_y^2 & xyI_y^2 \\ yI_xI_y & yI_y^2 & xyI_xI_y & y^2I_xI_y & xyI_y^2 & y^2I_y^2 \end{pmatrix} \tag{1.8}$$

$$\bar{d} = \sum_{\mathbf{x}\in\Re} \nabla I \delta I = \sum_{\mathbf{x}\in\Re} \begin{pmatrix} I_x \delta I \\ I_y \delta I \\ I_{xx} \delta I \\ I_{xy} \delta I \\ I_{yx} \delta I \\ I_{yy} \delta I \end{pmatrix} \tag{1.9}$$

В таком случае уравнение (1.7) можно переписать в виде

$$\frac{1}{2} D^T \approx \mathcal{G}\bar{v} - \bar{d} \tag{1.10}$$

Из условия оптимальности следует, что функция ошибки минимизируется при следующем значении параметров:

$$\bar{v}^{opt} = \begin{pmatrix} v_x^{opt} & v_y^{opt} & v_{xx}^{opt} & v_{xy}^{opt} & v_{yx}^{opt} & v_{yy}^{opt} \end{pmatrix} = \mathcal{G}^{-1}\bar{d} \tag{1.11}$$

В таком случае искомыми преобразованиями будут следующие:

$$b_{opt} = \begin{pmatrix} v_x^{opt} \\ v_y^{opt} \end{pmatrix}$$

$$A_{opt} = \begin{pmatrix} 1 + v_{xx}^{opt} & v_{xy}^{opt} \\ v_{yx}^{opt} & 1 + v_{yy}^{opt} \end{pmatrix}$$

Стандартное уравнение Лукаса–Канаде для определения оптического потока и аффинного преобразования (1.10) верно только для небольших смещений пикселей (чтобы гарантировать корректность разложения в ряд Тейлора). На практике, для получения точного решения данный вычислительный процесс необходимо повторять несколько раз. Далее описаны детали итеративного алгоритма в соответствии с [4].

Пусть k — порядковый номер очередной итерации. Опишем алгоритм рекурсивно: предположим, что вычисления на предыдущих итерациях $1, \ldots, k-1$ дали некоторое начальное приближение $b^{k-1} = \begin{pmatrix} v_x^{k-1} & v_y^{k-1} \end{pmatrix}^T$ и $A^{k-1} = \begin{pmatrix} 1 + v_{xx}^{k-1} & v_{xy}^{k-1} \\ v_{yx}^{k-1} & 1 + v_{yy}^{k-1} \end{pmatrix}$. Пусть $J_k(\mathbf{x})$ — это новое изображение, полученное из

исходного изображения $J(\mathbf{x})$ найденными преобразованиями $\{b^{k-1}; A^{k-1}\}$:

$$J_k(\mathbf{x}) = J(A^{k-1}\mathbf{x} + b^{k-1})$$

В таком случае цель очередной итерации — найти относительный сдвиг $\bar{\eta}^k = \begin{pmatrix} \eta_x^k & \eta_y^k \end{pmatrix}$ и относительное аффинное преобразование $\mathcal{M}^k = \begin{pmatrix} 1+\eta_{xx}^k & \eta_{xy}^k \\ \eta_{yx}^k & 1+\eta_{yy}^k \end{pmatrix}$, которые минимизируют функцию ошибки

$$\epsilon^k(\bar{\eta}^k, \mathcal{M}^k) = \epsilon^k(\eta_x^k, \eta_y^k, \eta_{xx}^k, \eta_{xy}^k, \eta_{yx}^k, \eta_{yy}^k) = \sum_{\mathbf{x}\in\mathfrak{R}} (I(\mathbf{x}) - J(\mathcal{M}^k\mathbf{x} + \bar{\eta}^k))^2 \quad (1.12)$$

Решение этой задачи может быть найдено с помощью одношагового метода Лукаса–Канаде (1.11):

$$\begin{pmatrix} \eta_x^k \\ \eta_y^k \\ \eta_{xx}^k \\ \eta_{xy}^k \\ \eta_{yx}^k \\ \eta_{yy}^k \end{pmatrix} = \mathcal{G}^{-1}\bar{d}_k \quad \Rightarrow \quad \begin{cases} \bar{\eta}^k = \begin{pmatrix} \eta_x^k \\ \eta_y^k \end{pmatrix} \\ \mathcal{M}^k = \begin{pmatrix} 1+\eta_{xx}^k & \eta_{xy}^k \\ \eta_{yx}^k & 1+\eta_{yy}^k \end{pmatrix} \end{cases} \quad (1.13)$$

где вектор $\bar{d}_k$ (вектор разности) определяется аналогично (1.9):

$$\bar{d}_k = \sum_{\mathbf{x}\in\mathfrak{R}} \begin{pmatrix} I_x\delta I_k \\ I_y\delta I_k \\ I_{xx}\delta I_k \\ I_{xy}\delta I_k \\ I_{yx}\delta I_k \\ I_{yy}\delta I_k \end{pmatrix}, \qquad \delta I_k(\mathbf{x}) = I(\mathbf{x}) - J_k(\mathbf{x}) \quad (1.14)$$

Матрица градиентов $\mathcal{G}$ при этом вычисляется единожды перед началом итеративного процесса.

После нахождения очередного относительного смещения и относительного аффинного преобразования, необходимо обновить значения A^k и b^k для

следующей итерации:

$$\begin{aligned} b^k &= b^{k-1} + A^{k-1}\bar{\eta}^k \\ A^k &= A^{k-1}\mathcal{M}^k \end{aligned} \tag{1.15}$$

Итерации повторяются до тех пор, пока относительный сдвиг $\bar{\eta}^k$ не станет меньше некоторого заданного порогового значения (например, 0,01 пикселя), или пока не будет выполнено некоторое максимальное количество итераций (если процесс не сходится).

В качестве начальных приближений для сдвига и аффинного преобразования можно взять известные приближения, или нейтральные преобразования в случае, если начальные приближения неизвестны:

$$b^0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$A^0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

1.2.1 Пирамидальная реализация алгоритма

При сравнительно больших смещениях есть риск, что итеративный метод сойдется к локальному минимуму, не достигнув глобального минимума (Рис. 1.2). Для того, чтобы избежать такого решения, а также для ускорения сходимости алгоритма была предложена пирамидальная реализация [4].

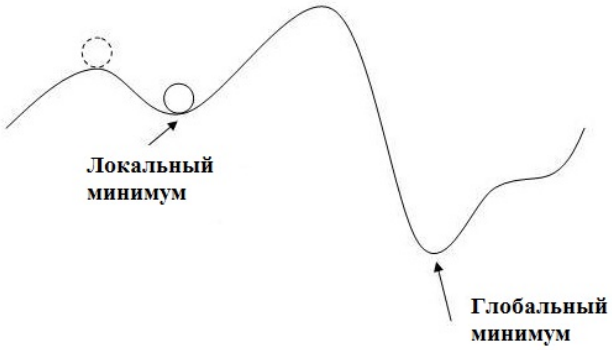


Рис. 1.2: Иллюстрация локального и глобального минимумов

Пирамидальное представление (Рис. 1.3) изображения I размера $n_x \times n_y$

определяется следующим образом[4]. Нулевым уровнем пирамиды считается само изображение $I^0 = I$. Размеры на этом уровне пирамиды, соответственно, равны $n_x^0 = n_x$, $n_y^0 = n_y$. Остальные уровни пирамиды строятся рекурсивно. Пусть $L = 1, 2, \ldots$ — уровень пирамиды, а I^{L-1} — пирамида уровня $L-1$. Обозначим через n_x^{L-1}, n_y^{L-1} размеры изображения I^{L-1}. Тогда изображение I^L будет определяться следующим образом:

$$I^L(x,y) = \frac{1}{4} I^{L-1}(2x, 2y) +$$
$$\frac{1}{8}(I^{L-1}(2x-1, 2y) + I^{L-1}(2x+1, 2y) + I^{L-1}(2x, 2y-1) + I^{L-1}(2x, 2y+1)) +$$
$$\frac{1}{16}(I^{L-1}(2x-1, 2y-1) + I^{L-1}(2x-1, 2y+1) + I^{L-1}(2x+1, 2y-1) + I^{L-1}(2x+1, 2y+1))$$

Для простоты обозначений, определим значения пикселей за пределами изображения на границе с ним ($0 \le x \le n_x^{L-1} - 1$, $0 \le y \le n_y^{L-1} - 1$):

$$I^{L-1}(-1, y) = I^{L-1}(0, y)$$

$$I^{L-1}(x, -1) = I^{L-1}(x, 0)$$
$$I^{L-1}(n_x^{L-1}, y) = I^{L-1}(n_x^{L-1} - 1, y)$$
$$I^{L-1}(x, n_y^{L-1}) = I^{L-1}(x, n_y^{L-1} - 1)$$
$$I^{L-1}(n_x^{L-1}, n_y^{L-1}) = I^{L-1}(n_x^{L-1} - 1, n_y^{L-1} - 1)$$

При этом размеры изображения I^{L-1} будут определяться следующими соотношениями:

$$n_x^L = \left[\frac{n_x^{L-1} + 1}{2}\right]$$
$$n_y^L = \left[\frac{n_y^{L-1} + 1}{2}\right]$$

Приведенные выше соотношения используются для построения пирамидальных представлений фиксированного I и подвижного J изображений: $\{I^L\}_{L=0,\ldots,L_m}$ и $\{J^L\}_{L=0,\ldots,L_m}$. Величина L_m, называемая высотой пирамиды, определяется, как правило, эвристически. На практике высоту выбирают

[4]Источник иллюстрации: `http://cs.brown.edu/courses/csci1430/2011/results/proj1/georgem/pyramid.gif`

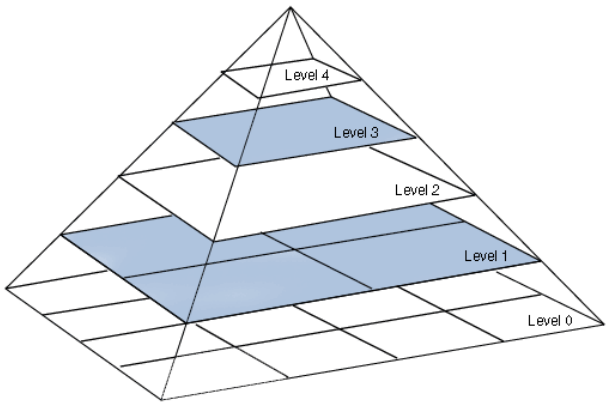


Рис. 1.3: Пирамидальное представление изображения

равной 2, 3 или 4, для типовых размеров регистрируемых изображений не имеет смысла брать высоту пирамиды больше четырех. Например, для изображения I размером 640×480, размерами изображений I^1, I^2, I^3 и I^4 будут, соответственно, 320×240, 160×120, 80×60 и 40×30. Главное преимущество пирамидального представления — возможность регистрировать большие сдвиги (больше размеров интересующего региона), поэтому величину высоты пирамиды следует выбирать также исходя из ожидаемого максимального сдвига.

Описанный в предыдущем разделе итеративный алгоритм регистрации применяется последовательно на каждом уровне пирамиды, начиная с самого верхнего I^{L_m}, J^{L_m} и заканчивая исходными изображениями I^0, J^0. При этом результат работы алгоритма на уровне L используется для вычисления начального приближения на уровне $L - 1$:

$$b^{L-1} = 2b^L$$

$$A^{L-1} = A^L$$

1.3 Сегментация на основе Марковской сети

Одной из задач, возникших в процессе проведения данного исследования, оказалась задача сегментации изображений, то есть разделения всех пикселей изображения на несколько категорий (сегментов), отвечающих определенным свойствам. Одна из распространенных задач сегментации (реша-

емая также в нашем исследовании) является задача разделения изображения на фон и объект. Одним из хорошо известных алгоритмов сегментации является алгоритм, основанный на Марковских сетях [21].
Марковская сеть (Марковское случайное поле) определяется следующими параметрами:

- Множество вершин $S = \{1, 2, \dots N\}$ (вершины соответствуют пикселям изображения)
- Множество случайных величин $\{w_n\}_{n=1}^{N}$, сопоставленных каждой вершине
- Множество соседей (окрестность) $\{\mathcal{N}_n\}_{n=1}^{N}$, заданная для каждой вершины

Чтобы быть Марковской сетью, эта структура должна обладать свойством марковости:

$$P(w_n|w_{S\setminus n}) = P(w_n|w_{\mathcal{N}_n}),$$

другими словами, значение случайной величины не зависит от всех остальных случайных величин, при известных соседях. Таким образом, вероятностное распределение случайных величин $\{w_i\}$ зависит от некоторого подмножества случайных величин $\mathcal{C}_i \subset \{1, 2, \dots, N\}$, называемого кликой.
Задача сегментации заключается в сопоставлении каждому пикселю одной

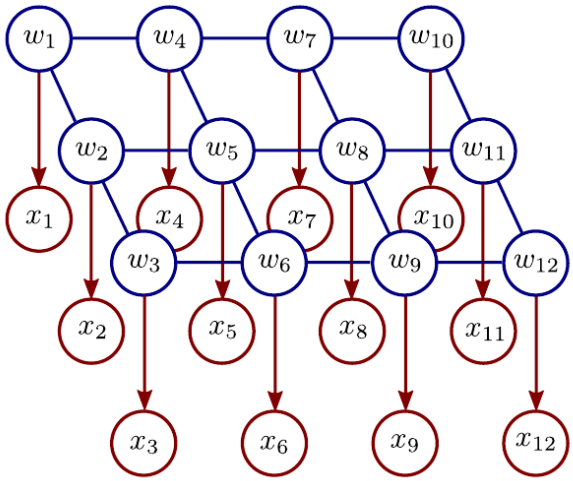


Рис. 1.4: Модель Марковской сети: w_i — состояния реального мира, x_i — наблюдаемые состояния

из меток $\{1, 2, 3, \dots, k\}$, где k — количество классов, на которые сегменти-

руется изображение. В нашей задаче участков будет два, метки, соответственно будут 0 — фон и 1 — объект. Задача сегментации сводится к задаче минимизации [21]

$$\bar{w}_{1\ldots N} = argmin \left[\sum_{i=1}^{N} U_i(w_i) + \sum_{(i,j)\in\mathcal{C}} P_{ij}(w_i, w_j) \right] \tag{1.16}$$

где $w_i \in \{1, 2, 3, \ldots, k\}, i = 1, \ldots, N$ — метки, сопоставляемые каждому из N пикселей изображения, $U_i(w_i)$ - стоимость назначения каждому пикселю той или иной метки, $P_{ij}(w_i, w_j)$ - стоимость назначения соответствующих меток соседним пикселям, входящим в одну клику $\mathcal{C}$ — задает гладкость сегментации (например, штрафуя за различные метки у соседних пикселей).

Для задачи (1.16) известно несколько решений, основанных на идее потоков в сетях — формируется взвешенный ориентированный граф с источником и стоком, весами которого являются функции стоимости $U_i(w_i)$ и $P_{ij}(w_i, w_j)$. В таком случае задача (1.16) сводится к задаче нахождения максимального потока и минимального разреза, для которой есть хорошо известные решения (например, алгоритм Форда-Фолкерсона). Этот алгоритм называется также max-flow/min-cut. Другой алгоритм, известный как алгоритм альфа–распространения, заключается в итеративном применении предыдущего алгоритма: формируется начальная разметка изображения, на каждой итерации выбирается очередная метка и решается бинарная задача - менять или нет метку каждого пикселя на новую. Каждая итерация данного алгоритма гарантированно уменьшает общую функцию стоимости, однако конечный результат может отличаться от глобального минимума.

В качестве программной реализации алгоритмов вывода на основе Марковских сетей была взята библиотека `GCoptimization` (библиотека минимизации энергии в разрезах сетей) [15, 6, 5], в которой реализуется алгоритм альфа–распространения.

Глава 2

Обзор работы

2.1 Постановка задачи

Задача данной работы заключается в следующем:

1. Исследовать влияние применения маски весов на качество регистрации изображений (преимущественно непрямоугольной формы на неоднородном фоне)
2. Разработать алгоритм формирования маски весов
3. Реализовать программную библиотеку регистрации изображений

2.2 Применение маски в алгоритме регистрации

Как отмечалось во введении, применение имеющихся реализаций данного алгоритма (например, ITK [13]) к изображениям со светлым фоном или со сдвигом объекта при неподвижном фоне дает плохие результаты. Причиной такого поведения является тот факт, что пиксели за пределами изображения по умолчанию считаются черными, что дает большую разность, и как следствие, большое значение функции ошибки (1.1), а также то, что при смещении неоднородного фона разность изображений также возрастает. В такой ситуации алгоритм Лукаса–Канаде, имеющий в своей основе идею градиентного спуска, предпочитает не делать смещений, чтобы минимизировать функцию ошибки.

Одним из решений этой проблемы является введение в алгоритм окна весов, которая позволит учитывать различные пиксели с различным весом, тем самым снизив влияние фоновых пискселей на минимизируемую сумму. Таким образом, задача регистрации будет сводиться к минимизации функции ошибки:

$$\epsilon(b, A) = \epsilon(v_x, v_y, v_{xx}, v_{xy}, v_{yx}, v_{yy}) = \sum_{\mathbf{x}\in\Re} W(\mathbf{x})(I(\mathbf{x}) - J(A\mathbf{x} + b))^2, \quad (2.1)$$

где W — некоторая весовая функция. При этом градиент (1.2) будет выглядеть так:

$$D = -2\sum_{\mathbf{x}\in\Re} W(\mathbf{x})(I(\mathbf{x}) - J(A\mathbf{x} + b))D_2(\mathbf{x}) \quad (2.2)$$

Решением этой задачи в таком случае будет являться вектор:

$$\bar{v}^{opt} = \begin{pmatrix} v_x^{opt} & v_y^{opt} & v_{xx}^{opt} & v_{xy}^{opt} & v_{yx}^{opt} & v_{yy}^{opt} \end{pmatrix} = (W\mathcal{G})^{-1}W\bar{d} \quad (2.3)$$

В качестве весовой функции была выбрана функция Ханна (Ханнинга), которое принимает максимальное значение в центре региона и плавно стремится у нулю на его краях (Рис.2.1). Такое поведение функции Ханна позволяет учитывать центральную часть окна с большим весом, а граничные регионы с меньшим, снижая воздействие фона на регистрацию подвижного объекта. В то же время функция Ханна сравнительно проста по своей форме.

$$W(n) = 0.5\left(1 - cos\left(\frac{2\pi n}{N-1}\right)\right) \quad (2.4)$$

Для наших целей данна функция была модифицирована:

$$W(\mathbf{x}) = 0.5\left(1 - cos\left(\frac{\pi R(\mathbf{x})}{R_{max}}\right)\right), \quad (2.5)$$

где $R(\mathbf{x})$ — расстояние от пикселя до границы региона, а R_{max} — максимально возможное расстояние от пикселя до границы региона (Рис. 2.2). Для большей точности предлагается использовать маску, совпадающую по контурам с регистрируемой областью, так как такой подход позволяет не

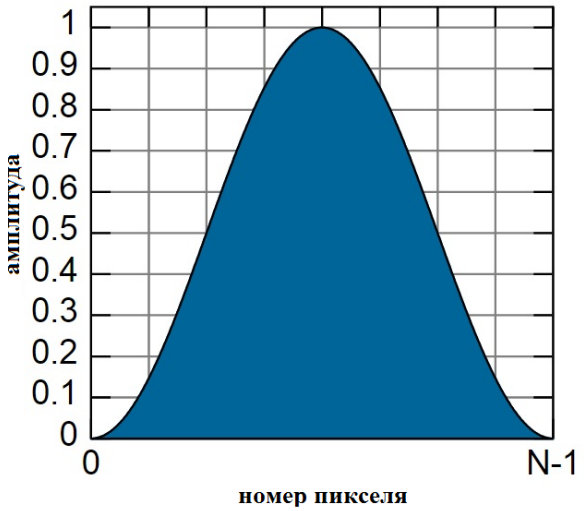


Рис. 2.1: Одномерное окно Ханна

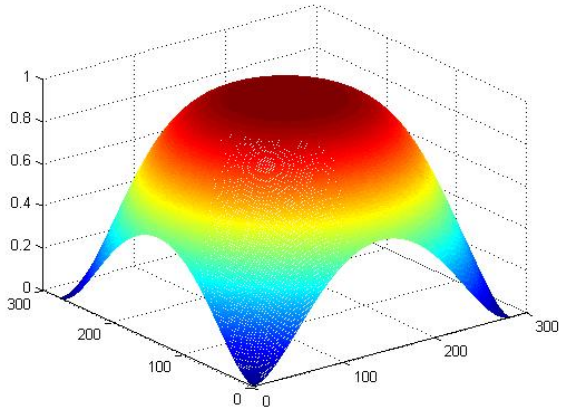


Рис. 2.2: Двумерное окно Ханна

терять информацию, содержащуюся в изображении регистрируемого объекта, максимально ослабляя влияние фона на минимизируемую функцию ошибки (2.1).

Для экспериментов были сгенерированы черно-белые изображения следующим образом: выбирался неоднородный фон и нерегулярный объект (не обладающий прямоугольной формой, неоднородный); для «неподвижного» изображения объект накладывался на фон в исходном виде, для «движущегося» изображения объект подвергался известным преобразованиям (поворот, растяжение/сжатие, наклон) и накладывался на фон в смещенной по отношению к «неподвижному» изображению позиции (с известным смещением) (Рис. 2.3). Для сравнительного эксперимента была проведена серия регистраций для известных сдвигов (1, 2, 3, 5, 15) пикселей по одной

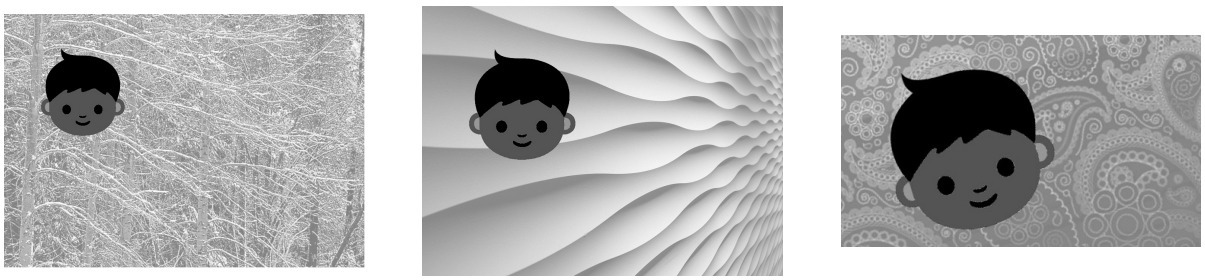
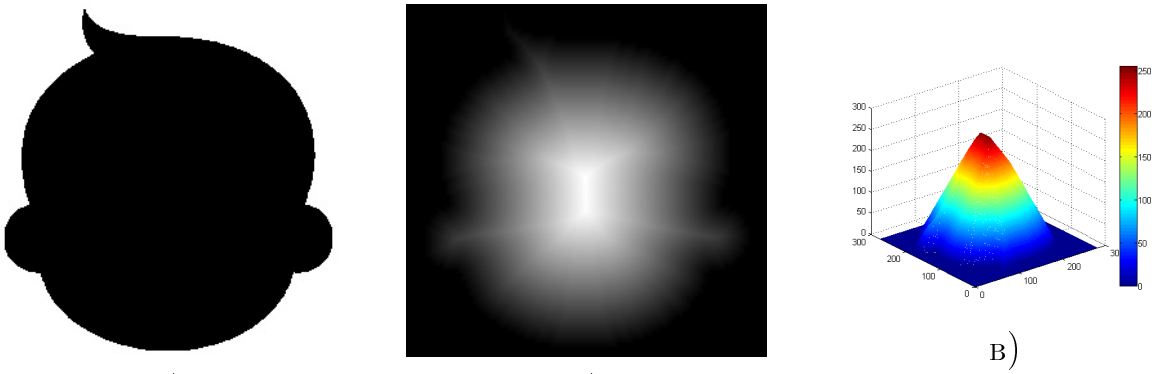


Рис. 2.3: Примеры сгенерированных тестовых изображений.

или нескольким осям. Регистрация проводилась стандартными алгоритмом Лукаса–Канаде и алгоритмом с использованием маски и окна весов.
Окно весов формировалось следующим образом: генерировалась маска по контуру регистрируемого объекта таким образом, чтобы пиксели, принадлежащие объекту были белого цвета (имели значение 255 в одноканальном черно-белом изображении), а пиксели фона имели черный цвет (значение 0 в одноканальном черно-белом изображении) (Рис. 2.4а). Затем к полученной маске применяется функция `distanceTransform` библиотеки компьютерного зрения `openCV`[1]. Данная функция формирует матрицу, в которой каждому белому пикселю ставится в соответствие кратчайшее расстояние до границы белой области, к которой он принадлежит, а всем пикселям за пределами светлых областей ставится в соответствие 0 (Рис. 2.4б). Последним этапом к полученной матрице расстояний применяется модифицированная функция Ханна (2.5) (Рис. 2.4в). Для сравнения вы-



а) б) в)

Рис. 2.4: Маска и окно весов: а) маска объекта; б) матрица расстояний от пикселей до границы (чем светлее пиксель, тем больше расстояние); в) окно Ханна по форме маски (нормализовано до диапазона 0–255 для наглядности)

числялась средняя относительная ошибка сдвига (модуль ошибки вектора

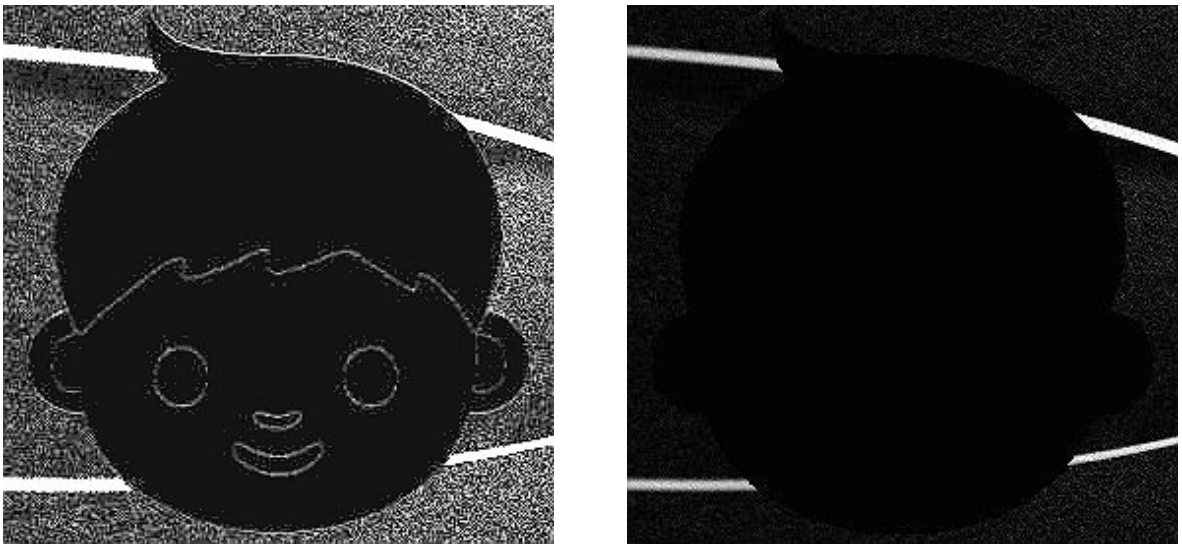
[1] `http://opencv.org`

свига в сравнении с известным вектором сдвига) и среднеквадратичное отклонение исходного фиксированного изображения и изображения, полученного из "движущегося"преобразованием, найденным в процессе регистрации. Результаты приведены в таблице 2.2. На рисунке 2.5 изображены

Метод регистрации	Средняя относительная ошибка сдвига	Среднеквадратичное отклонение изображений
Без оконной функции	4%	68.51
С использованием оконной функции	$13 \cdot 10^{-5}\%$	$7.08 \cdot 10^{-7}$

Таблица 2.1: Сравнение результатов регистрации с использованием оконной функции и без нее

примеры разностей изображений —- исходного и полученного в процессе регистрации. Как видно на рис. 2.5а, сдвиг найден недостаточно точно, в результате чего наложение изображений не идеальное, и на объекте контуры линий. На изображении 2.5б объект полностью черный, что означает, что исходное и полученное изображения наложились точно, и разность соответствующих пикселей оказалась равна нулю. Таким образом, видно, что



а) б)

Рис. 2.5: Разности исходного и найденного в процессе регистрации изображений: а) без использования окна весов; б) с использованием окна весов, построенных по известной маске (в изображения добавлена контрастность для наглядности)

применение весовой функции, построенной на основе известной маски, совпадающей по контурам с регистрируемым объектом, значительно улучшает точность регистрации изображений. Очевидная сложность такого подхода заключается в том, что в обычных приложениях точная форма объекта неизвестна, либо требует применения не менее сложных алгоритмов для определения, как следствие, невозможно построить маску до выполнения регистрации изображений. Следующий раздел посвящен решению вопроса формирования маски.

2.3 Алгоритм регистрации с адаптивной маской

Основная идея алгоритма построения адаптивной маски заключается в попытке сегментации изображения по признаку фон–объект, а затем использовании сегментированного изображения для построения соответствующей матрицы весов.

Так как регистрируемые объекты как правило представляют собой связные области, для разметки фон–объект можно использовать алгоритм альфа–распространения или алгоритм max–flow/min–cut в Марковских сетях, как было показано в предыдущем разделе. Однако для применения алгоритма сегментации на основе Марковской сети, нужно задать некоторое начальное распределение меток фон–объект.

Для начальной сегментации можно использовать эвристический подход, основанный на идее, что при сравнении исходного изображения и найденного в процессе регистрации, регистрируемые объекты наложатся друга на друга сравнительно точно, а фон окажется сдвинутым. Как следствие, пиксели объекта будут иметь в среднем меньшую разность, чем пиксели фона (см. рис. 2.5).

Для определения начальной сегментации предлагается Алгоритм 1.

Выбор упомянутого эмпирического порога может отличаться от случая к случаю: если фон на изображении неоднородный, то можно ожидать, что пиксели фона будут иметь значительно большее среднеквадратичное отклонение, чем пиксели однородных областей объекта, и порог можно выбирать высоким, и обратно, при достаточно однородном фоне большое ко-

Algorithm 1 Алгоритм формирования маски

1: Разбить исходное «движущееся» изображение на блоки 3×3 пикселя
2: К каждому блоку применить найденное в процессе регистрации аффинное преобразование и посчитать средневкадратичное отклонение от «неподвижного» изображения
3: Пиксели в тех блоках, отклонение в которых меньше некоторого эмпирически определяемого порога, будем считать принадлежащими объекту, остальные — принадлежащими фону

а) б)

Рис. 2.6: Сформированные маски (белые пиксели — объект, черные — фон): а) после применения эмпирического порога (заметно значительное количество пикселей, помеченных неверно); б) после применения алгоритма сегментации на основе Марковских сетей (число неверно помеченных пикселей уменьшилось)

личество пикселей фона в случае высокого порога будет помечено как принадлежащие объекту. Эмпирический порог в экспериментах был выбран равным среднеквадратичному отклонению пикселей по всему изображению целиком; данный порог был определен в результате экспериментов со сгенерированными изображениями (например, см. рис. 2.3) и давал хорошие результаты в случае реальных изображений. В этом случае часть пикселей, которые на самом деле принадлежат фону, помечаются как пиксели объекта (это особенно характерно для фона с областями однородности или периодичности), что можно видеть на рис. 2.6а. Кроме того, часть пикселей объекта помечаются как фоновые из-за неточностей найденного преобразования, и как следствие сравнительно большой разницы (на грани-

цах областей однородности). Чтобы решить эту проблему, дальнейшим этапом формирования маски является применение алгоритма сегментации на основе Марковской сети (max–flow/min–cut или альфа-распространение). Данный алгоритм за счет локальности марковского свойства «сглаживает» ошибочно размеченные регионы, уменьшая число неверных меток (см. рис. 2.6б). Оставшиеся ошибочные регионы за пределами объекта, помеченные, как принадлежащие объекту, существенной роли в регистрацию вносить тем не менее не будут из-за своего малого размера и выбранной весовой функции (2.5): так как для небольшого региона расстояние R от его пикселей до его границы значительно меньше максимального расстояния R_{max}, вычисляемого по всей маске, то функция Ханна (2.5) для этих пикселей будет давать значения, близкие к нулю.

Описанный выше алгоритм формирования маски предполагает, что аффинное преобразование между «неподвижным» и «движущимся» изображениями известно, то есть процесс регистрации уже осуществлен. Вместе с тем, было бы нерационально применять алгоритм регистрации дважды — первый раз, чтобы найти начальное приближение геометрического преобразования, а затем еще раз, с уже сформированной маской.

Для решение этой задачи предлагается формировать маску объекта на каждом уровне пирамиды, используя приближения параметров аффинного преобразования, найденные на предыдущем этапе. Этот подход позволит уточнять маску поэтапно, одновременно повышая точность регистрации на каждом этапе. Чтобы избежать упомянутых уже эффектов светлого или подвижного фона, на самом высоком уровне пирамиды предполагается использовать фиксированную матрицу весов (функцию Ханна, определенную по всему регистрируемому региону, см. рис. 2.2).

Таким образом, из описанной модификации алгоритма Лукаса–Канаде с использованием окна весов и алгоритма формирования маски объекта получается алгоритм регистрации с адаптивной маской (Алгоритм 2).

2.4 Программная реализация

Для проведения экспериментов была создана библиотека на языке программирования C++, в качестве среды разработки использовалась IDE

Algorithm 2 Алгоритм регистрации с адаптивной маской

Дано: фиксированное и подвижное изображения I и J
Сформировать пирамидальные представления $\{I^L\}_{L=0,\ldots,L_m}$ и $\{J^L\}_{L=0,\ldots,L_m}$
$A_g = ones(2,2); b_g = zeros(1,2)$, или начальные приближения, если известны
Инициализация регистрации: $A = A_g$, $b = b_g/2^{L_M+1}$
for $L = L_m$ downto 0 **do**
    if не установлена маска **then**
        Установить простую маску, каждому пикселю ставится в соответствие расстояние до границ региона
        Вычислить матрицу весов W(x) по формуле (2.5)
    end if
    Вычисление частных производных по формулам (1.5), (1.6)
    Нахождение матрицы градиентов $\mathcal{G} = \sum_{\mathbf{x}\in\mathfrak{R}} W(\mathbf{x})\nabla I \nabla I^T$
    Инициализация начального приближения сдвига с предыдущего уровня пирамиды $b = 2b$
    repeat
        Преобразование подвижного изображения $J = J^L(Ax + b)$
        Вычисление несоответствия изображений $\delta I(\mathbf{x}) = W(\mathbf{x})(I(\mathbf{x}) - J(\mathbf{x}))$
        Вычисление вектора разности $\bar{d} = \sum_{\mathbf{x}\in\mathfrak{R}} \nabla I \delta I$
        Решение $\bar{v}^{opt} = \mathcal{G}^{-1}\bar{d}$ с помощью SVD-разложения
        Обновление решения $b = b + A\bar{\eta}$, $A = A\mathcal{M}$, см. (1.15)
    until $|\eta| < \epsilon$ или $iteration > MaxIteration$
    if L > 0 **then**
        Применить найденное аффинное преобразование ($b \leftarrow 2b$) к $L-1$ уровню пирамиды J^{L-1}
        Разбить фиксированное изображение I^{L-1} на блоки 3×3
        Для каждого блока посчитать среднеквадратичное отклонение от соответствующего блока преобразованного изображения J^{L-1}
        Если отклонение меньше 0,1, установить для всех пикселей блока метку 1, иначе 0
        Применить алгоритм альфа-распространения для начальной разметки
        Сформировать изображение размером $n_x^{L-1} \times n_y^{L-1}$, установить значения пикселей 0, если метка 0, и 255, если метка 1
        Применить к полученному изображению метод `distanceTransform`
        Вычислить матрицу весов $W(x)$ по формуле (2.5)
        Установить найденные маску и матрицу весов для следующей итерации цикла
    end if
end for

Visual Studio 2012. Реализованная библиотека использует сторонние программные средства с открытым кодом: библиотеку `OpenCV` и библиотеку вывода на основе Марковских сетей `GCoptimization` ([15, 6, 5]).

2.4.1 Описание классов библиотеки

Библиотека состоит из трех классов:

- Класс `Image` представляет собой обертку для класса `Mat` библиотеки `OpenCV`, обеспечивает возможность работы с изображением, как с непрерывной функцией
- Класс `ImageRegistrationLK` — основной класс библиотеки, осуществляющий регистрацию изображений
- Класс `MRFSegmentation` обеспечивает возможность сегментации изображений на основе Марковских сетей, используется в процессе регистрации для формирования маски

Класс `Image` абстрагирует изображение от представления в виде матрицы до представления в виде непрерывной функции двух переменных. Класс предоставляет открытый метод `at(double x, double y)`, возвращающий значение двумерной функции в точке `(x, y)`. Значение функции вычисляется с помощью билинейной интерполяции, при этом считается, что значения пикселей за пределами изображения равны нулю. Открытый метод `setTransform(Mat A, Mat d)` устанавливает для изображения некоторое аффинное преобразование с матрицей преобразования `A` и вектором сдвига `d`. Переданные в качестве параметров преобразования хранятся внутри объекта и применяются к координатам точек по мере необходимости, то есть актуального преобразования матрицы изображения не происходит. Также класс предоставляет интерфейс для получения градиента изображения в заданной точке `grad(int x, int y)`. Данный класс активно используется в основном классе регистрации.

Класс `ImageRegistrationLK` является основным классом библиотеки и предоставляет возможность регистрировать два изображения. Для задания параметров класса можно использовать следующие методы:

- `setFixedImage(Mat)` — определяет «неподвижное» изображение для регистрации
- `setMovingImage(Mat)` — определяет «движущееся» изображение для регистрации
- `setWindowSize(Size)` — задает размер регистрируемого региона
- `setWindowOffset(Point2d)` — задает смещение регистрируемого региона относительно левого верхнего угла изображения
- `setNumberOfLevels(int)` — задает количество уровней пирамиды (по умолчанию 2)
- `setPrecision(double)` — задает необходимую точность для сходимости итеративного процесса (по умолчанию 0.00001)
- `setMaxIterations(int)` – задает максимальное число итераций (по умолчанию 25)
- `setMask(Mat)` — задает маску, если она известна заранее
- `setRegistrationMode(int)` — задает геометрическое преобразование, которое ищется в процессе регистрации

Функция `setRegistrationMode(int)` позволяет задать один или несколько режимов регистрации (искомых геометрических преобразований) с помощью флагов из следующего набора:

- `REG_SHIFT` — параллельный сдвиг
- `REG_ROTATION` – поворот
- `REG_SKEW` — скос
- `REG_SCALE` — масштабирование по осям
- `REG_UNIFORM_SCALE` — однородное масштабирование
- `REG_AFFINE` — произвольное аффинное преобразование

После настройки необходимых параметров алгоритма регистрации, процесс запускается вызовом метода `runRegistration()`. После завершения регистрации найденное преобразование можно узнать с помощью метода `getTransform()`, который вернет матрицу аффинного преобразования, которую затем можно применять в методе `warpAffine` библиотеки `OpenCV`. Достигнутую в результате регистрации разность изображений можно получить с помощью метода `getDifference()`.

Вспомогательный класс `MRFSegmentation` является оберткой для методов библиотеки `GCoptimization` и предоставляет только один метод `patternSegmentation(Mat, int)`, параметрами которого являются изображение и количество меток (в процессе регистрации второй параметр всегда равен 2, так как задача состоит в разбиении на фон и объект). Данный класс используется в процессе регистрации.

Библиотека состоит из трех заголовочных файлов `Image.h`, `ImageRegistrationLK.h`, `MRFSegmentation.h` и библиотечного файла `ImageSegmentation.lib`. Открытый код библиотеки доступен на публичном сервисе контроля версий GitHub[2]. Планируются различные улучшения библиотеки для возможности применять ее в сторонних приложениях.

Объетивным недостатком программной реализации является длительное время работы и отсутствие оптимизации по времени и по памяти. На начальном этапе работы основной задачей была проверка концепции и сравнительного анализа предложенного алгоритма с ранее известными версиями, поэтому задача оптимизации кода не ставилась. В дальнейшем предполагается улучшение производительности кода, в том числе за счет внедрения параллелизма (выполняемые в алгоритме операции хорошо подвергаются распараллеливанию).

[2] `https://github.com/maksimbolonkin/LK`

Глава 3

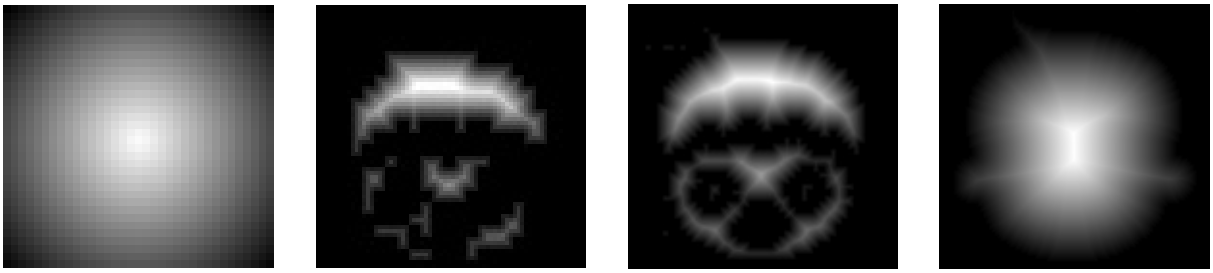
Результаты экспериментов

Для проверки работы алгоритмов была проведена серия численных экспериментов с использованием реализованной библиотеки регистрации изображений. Часть экспериментов была проведена с синтетическими изображениями (сформированными специально для целей эксперимента), а часть — на основе реальных кадров.

3.1 Эксперименты с синтетическими данными

Для проведения экспериментов были сформированы синтетические изображения из неоднородного фона и нерегулярного объекта. Так как одним из основных недостатков известных реализаций алгоритма Лукаса–Канаде является неудовлетворительная работы в случае светлого фона, фон искуственно осветлялся (нормализовывался до диапазона 128–255), а затем на полученный фон накладывалось изображене объекта. Для "неподвижного"изображения объект накладывался на фон в исходном виде, для "движущегося"изображения объект подвергался известным преобразованиям (поворот, растяжение/сжатие) и накладывался на фон в смещенной по отношению к "неподвижному"изображению позиции (с известным смещением) (Рис. 2.3). Для проверки работы алгоритма была проведена серия численных экспериментов с регистрацией известных сдвигов. Проверялось, помимо прочего, формирование маски изображения. На рисунке 3.1 представлена последовательность масок, которые используютя на каждом уровне

пирамиды: 3.1а — простая функция Ханна, примененная ко всему региону, эта маска используется как начальное приближение на самом высоком уровне пирамиды, когда ничего не известно про относительные геометрические преобразования "неподвижного"и "движущегося"изображений; 3.1б, 3.1в — на последующих этапах маска уточняется, видно, что контуры маски совпадают с контурами изображения за исключением некоторых артефактов; 3.1г — на последнем уровне найденное приближение уже достаточно хорошее, чтобы контуры маски достаточно точно совпадали с регистрируемым объектом.



а) б) в) г)

Рис. 3.1: Маски, получаемые на различных уровнях пирамиды: а) на 3 уровне (самом высоком); б) 2 уровень; в) 1 уровень; г) 0 уровень

Для сравнительного эксперимента была проведена серия регистраций для известных сдвигов (1, 2, 3, 5, 15) пикселей по одной или нескольким осям. Регистрация проводилась стандартными алгоритмом Лукаса–Канаде, алгоритмом Лукаса-Канаде с функций Ханна без маски (на каждом уровне пирамиды применялась функция Ханна по всему региону) и алгоритмом с адаптивной маской (Алгоритм 2).

Для сравнения вычислялась средняя относительная ошибка сдвига (модуль ошибки вектора свига в сравнении с известным вектором сдвига) и среднеквадратичное отклонение исходного фиксированного изображения и изображения, полученного из "движущегося"преобразованием, найденным в процессе регистрации. Результаты приведены в таблице 3.1

Таким образом, показано, что для сформированных изображений адаптивный алгоритм решает задачу регистрации лучше, чем алгоритм без использования маски. В некоторых случаях полученная маска отличается по контурам от регистрируемого объекта (зависит от фона, на котором расположен объект, и сложности формы объекта), однако во всех случаях получаемая маска покрывает значительную часть (больше 75% площади)

Метод регистрации	Средняя относительная ошибка сдвига	Среднеквадратичное отклонение изображений
Без оконной функции	4%	68.51
С использованием окна Ханна	0.87%	23.24
С использованием адаптивной маски	0.0025%	$9.12 \cdot 10^{-5}$

Таблица 3.1: Сравнение результатов регистрации различными алгоритмами

объекта, что позволяет улучшать процесс регистрации и добиваться улучшения точности регистрации.

3.2 Эксперименты с реальными данными

Кроме синтетически данных алгоритм проверялся на реальных кадрах видеотреков, снятых с помощью камеры или видеорегистратора.

На нескольких наборах кадров одного видео ставилась задача отслеживания какого-либо объекта (автомобиля, автобуса, фар и номерного знака). Задача решалась последовательным применением алгоритма регистрации к парам последовательных кадров, определялся масштаб и сдвиг, которые затем использовались для вычисления размеров окна регистрации и сдвига левого верхнего угла этого окна относительно начала координат всего изображения на следующей паре кадров. Размер и сдвиг окна на самом первом кадре определялись вручную.

На каждом наборе тестов были запущены три алгоритма — алгоритм регистрации Лукаса–Канаде, алгоритм регистрации с использованием окна Ханна по всему региону регистрации и алгоритм регистрации с адаптивным окном.

На первом наборе кадров была сделана попытка отследить автомобиль, перемещающийся по трассе по направлению от наблюдателя[1]. Из Рис. 3.2 можно увидеть, что в случае регистрации алгоритмом Лукаса–Канаде без

[1]Данные кадры были взяты из видеотрека `http://www.videezy.com/transportation/188-free-cars-driving-at-night-stock-video-in-hd`

использования окон машина не отслеживается вообще. Возможная причина такого поведения — сравнительно яркое освещение участка дороги, на котором находится в этот момент автомобиль. Алгоритм регистрации с использованием окна Ханна по всему региону регистрации начинает отслеживание, однако отслеживается небольшой участок рядом с машиной, затем поисходит потеря объекта (Рис. 3.3). В то же время алгоритм с использованием адаптивного окна довольно долго отслеживает нужный автомобиль (Рис. 3.4) и только на последних кадрах начинается выход автомобиля из региона отслеживания, что может быть связано с появлением геометрических преобразований, отличных от сдвига и масштабирования.

Рис. 3.2: Регистрация алгоритмом Лукаса–Канаде

Рис. 3.3: Регистрация алгоритмом с использованием окна Ханна по всему региону регистрации

Рис. 3.4: Регистрация алгоритмом с использованием адаптивного окна

В то же время на другом тестовом наборе кадров алгоритм с адаптивной маской и стандартный алгоритм Лукаса–Канаде дают сходные друг с другом результаты. Отслеживался автобус, который перемещался на неоднородном фоне, а на переднем плане в противоположную сторону перемещался автомобиль. На рисунках 3.6 и 3.5 представлены отслеженные регионы — легко видеть, что в обоих случаях регистрируются одни и те же регионы изображения.

Рис. 3.5: Регистрация алгоритмом Лукаса–Канаде без использования окна

Рис. 3.6: Регистрация алгоритмом с использованием адаптивного окна

Таким образом, можно сделать вывод, что алгоритм регистрации с адаптивной маской работает не хуже, чем стандартный алгоритм регистрации Лукаса–Канаде, а в некоторых случаях дает существенно лучшие результаты.

Глава 4

Заключение

В рамках работы над данной диссертацией была выполнены следующие подзадачи:

- Изучена задача повышения разрешения и имеющиеся решения этой проблемы
- Изучены различные средства регистрации изображений, имеющиеся в открытом доступе, рассмотрены преимущества и недостатки этих программных средств
- Подробно изучен и реализован программно алгоритм регистрации Лукаса–Канаде
- Предложена версия алгоритма Лукаса–Канаде с применением маски и окна весов, проведены эксперименты
- Разработан алгоритм формирования маски
- Реализован алгоритм регистрации с адаптивной маской в виде программной библиотеки, проведены эксперименты

К результатам работы можно отнести следующее:

- Предложена версия алгоритма Лукаса–Канаде с применением маски и окна весов, проведены эксперименты, показано, что применение известной маски улучшает точность регистрации

- Разработан адаптивный алгоритм формирования маски, проведены эксперименты, показавшие, что применение алгоритма регистрации с адаптивной маской в некоторых случаях улучшает точность регистрации, в остальных дает результат не хуже, чем обычный алгоритм регистрации

- Реализован алгоритм регистрации с адаптивной маской в виде программной библиотеки

Дальнейшая работа возможна в нескольких направлениях: оптимизация программного продукта, в том числе внедрение параллелизма, так как осуществляемые в алгоритме операции хорошо подвергаются распараллеливанию; улучшение процесса сегментации и переход от эмпирической границы в алгоритме формирования маски к границе, определяемой на основе полученных разностей.

Благодарности

В заключении автор хочет выразить благодарность своему научному руководителю к.ф.-м.н, доценту Вахитову Александру Тимуровичу за его неоценимый вклад в работу, терпение и поддержку, а также старшему преподавателю к.ф.-м.н Луциву Дмитрию Вадимовичу за его поддержку и помощь на протяжение всего периода обучения на магистерской программе.

Литература

[1] I. Boada A. Bardera, M. Feixas. Normalized similarity measures for medical image registration. *Medical Imaging SPIE*, 2004.

[2] J. Ashburner, P. Neelin, D.L. Collins, A. Evans, and K. Friston. Incorporating prior knowledge into image registration. *NeuroImage*, 6(4):344 – 352, 1997.

[3] Daniel I. Barnea and H.F. Silverman. A class of algorithms for fast digital image registration. *Computers, IEEE Transactions on*, C-21(2):179–186, Feb 1972.

[4] Jean-Yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000.

[5] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, September 2004.

[6] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, November 2001.

[7] Lisa Gottesfeld Brown. A survey of image registration techniques. *ACM Comput. Surv.*, 24(4):325–376, December 1992.

[8] G.K. Chantas, N.P. Galatsanos, and N.A. Woods. Super-resolution based on fast registration and maximum a posteriori reconstruction. *Image Processing, IEEE Transactions on*, 16(7):1821–1830, July 2007.

[9] Kevin W. Eliceiri, Michael R. Berthold, Ilya G. Goldberg, Luis Ibanez, B. S. Manjunath, Maryann E. Martone, Robert F. Murphy, Hanchuan

Peng, Anne L. Plant, Badrinath Roysam, Nico Stuurman, Jason R. Swedlow, Pavel Tomancak, and Anne E. Carpenter. Biological imaging software tools. *Nature Methods*, 9:697–710, 2012.

[10] Chong Fan, Jiangjun Zhu, Jianya Gong, and Cuiling Kuang. Pocs super-resolution sequence image reconstruction based on improvement approach of keren registration method. In *Intelligent Systems Design and Applications, 2006. ISDA '06. Sixth International Conference on*, volume 2, pages 333–337, Oct 2006.

[11] Manuel Guizar-Sicairos, Samuel T. Thurman, and James R. Fienup. Efficient subpixel image registration algorithms. *Opt. Lett.*, 33(2):156–158, Jan 2008.

[12] Yu He, Kim-Hui Yap, Li Chen, and L.-P. Chau. A nonlinear least square technique for simultaneous image registration and super-resolution. *Image Processing, IEEE Transactions on*, 16(11):2830–2841, Nov 2007.

[13] Luis Ibanez, Lydia Ng, J Gee, and S Aylward. Registration patterns: the generic framework for image registration of the insight toolkit. In *Biomedical Imaging, 2002. Proceedings. 2002 IEEE International Symposium on*, pages 345–348. IEEE, 2002.

[14] Michal Irani and Shmuel Peleg. Improving resolution by image registration. *CVGIP: Graph. Models Image Process.*, 53(3):231–239, April 1991.

[15] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:65–81, 2004.

[16] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[17] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens.

Multimodality image registration by maximization of mutual information. *Medical Imaging, IEEE Transactions on*, 16(2):187–198, April 1997.

[18] J.B.Antoine Maintz and Max A. Viergever. A survey of medical image registration. *Medical Image Analysis*, 2(1):1 – 36, 1998.

[19] Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. Super-resolution image reconstruction: a technical overview. *Signal Processing Magazine, IEEE*, 20(3):21–36, May 2003.

[20] Josien P.W. Pluim, J.B. Antoine Maintz, and Max A. Viergever. Interpolation artefacts in mutual information-based image registration. *Computer Vision and Image Understanding*, 77(2):211 – 232, 2000.

[21] Simon J. D. Prince. *Computer Vision: Models, Learning, and Inference*. Cambridge University Press, New York, NY, USA, 1st edition, 2012.

[22] Feng-qing Qin, Xiao-hai He, Wei-long Chen, Xiao-min Yang, and Wei Wu. Video superresolution reconstruction based on subpixel registration and iterative back projection. *Journal of Electronic Imaging*, 18(1):013007–013007–11, 2009.

[23] M. Dirk Robinson, Sina Farsiu, and Peyman Milanfar. Optimal registration of aliased images using variable projection with applications to super-resolution. *Comput. J.*, 52(1):31–42, 2009.

[24] Barbara Zitova and Jan Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977 – 1000, 2003.