САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет

Кафедра системного программирования

Жуков Роман Сергеевич

Экспериментальное исследование алгоритма оценивания положения движущейся точки

Курсовая работа

Научный руководитель:
доц. А. Т. Вахитов

Санкт-Петербург
2015

Оглавление

Введение

Робототехника применяется в промышленности (роботизированные конвееры), военном деле [10] (беспилотные аппараты, роботы для разминирования), автомобилестроении [5] (автономное вождение, разработанное компаниями Google, Daimler и др.) и в других областях. Если рассматриать человека как кибернетическую систему, то большую часть информации об окружающем мире ему приносит зрение. В случае роботов, можно ожидать аналогичного решения, поэтому тема зрения для роботов является очень актуальной.

Роботехнической системе необходимо ориентироваться в окружающем пространстве. В компьютерном зрении эта проблема называется одновременной навигацией и составлением карты (SLAM от англ. Simultaneous Localization and Mapping).

Существует множество решений SLAM [6], [11], [9]. Все они используют различные датчики и работают в различных условиях. Одни используют одну камеру и работают только в статической сцене, при этом отфильтровывая движущиеся объекты. Другие могут работать в динамической сцене, но при этом требуют либо дополнительных датчиков, например, радаров или дополнительных камер, либо ограничений перемещения объекта. Первый случай не интересен для рассмотрения, потому что использование дополнительных устройств может оказаться слишком дорогим вариантом для многих приложений. Во втором случае алгоритмы требуют достаточно больших ограничений. Например, в [9] требуется, чтобы объект двигался в одной плоскости с известным до нее расстоянием. А в [11] используются особенности, характерные только для области применения данного алгоритма. Такие ограничения сильно уменьшают практическое использование алгоритмов. В связи с этим было решено рассмотреть алгоритм восстановления глубины точки при наблюдении с одной камеры с помощью рандомизации ее положения, описанный в работе [4], который не имел вышеперечисленных недостатков. Для отслеживания объекта, двигающегося произвольно, требовалась всего одна камера.

Исследуемый алгоритм является рандомизированным алгоритмом стохастической оптимизации. Это означает, что для его работы необходимы случайные возмущения положения камеры. Следовательно необходимо отслеживать эти перемещения. Для решения этой проблемы требуется алгоритм определения положения камеры.

В следующих частях излагаются исследуемые алгоритмы и проведенные эксперименты для их апробации.

1. Постановка задачи

Целью работы является проведение экспериментов с роботами для исследования алгоритма [4], так как до этого в реальных условиях данный алгоритм еще не тестировался. В рамках данной работы были поставлены следующие задачи:

- Составить и реализовать алгоритм определения положения камеры
- Провести эксперименты с одним роботом для его тестирования
- На основе статьи [4] реализовать алгоритм оценки глубины движущейся точки
- Провести эксперименты с двумя движущимися роботами

2. Описание алгоритмов

Для решения поставленных задач была составлена система, включающая в себя все необходимые алгоритмы, схема работы которой выглядит следующим образом:

- На вход системы поступает изображение
- Детектируются точки неподвижного объекта
- Определяется положение камеры относительно неподвижного объекта
- Детектируется движущийся объект, положение которого необходимо определить
- Вычисляется положение движущегося объекта

Случайные возмущения положения камеры могут отслеживаться либо сторонним датчиком, либо по изображению известного неподвижного объекта. В проведенных экспериментах была выбрана вторая стратегия в силу того, что она является более простой в реализации, не требуя привлечения сенсоров кроме уже имеющейся камеры.

2.1. Детекция точек неподвижного объекта

Существует несколько способов нахождения ключевых точек: использование специальных методов таких, как SIFT, либо использование легко распознаваемых маркеров. Первый вариант более затратный, так как ключевых точек может оказаться слишком много, и тогда придется применять фильтрацию. Поэтому был выбран вариант с маркером в виде шахматной доски.

На первом кадре видео с камеры определяются углы шахматной доски. Рассмаривается система координат, привязанная к шахматной доске: левый верхний угол имеет координаты $(0, 0)$, а правый нижний – (n_x, n_y), где n_x и n_y - количество клеток по горизонтали и вертикали соответственно. Затем с помощью методов, реализованных библиотеке OpenCV [8], рассчитывается мартица гомографии H из плоскости доски в плоскость сенсора камеры, которая удовлетворяет следующей формуле проектированя в однородных координатах:

$$s \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K(r_1\ r_2\ r_3\ t) \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} = K(r_1\ r_2\ t) \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = H \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix},$$

где $(u, v)^T$ – координаты точки на изображении, $(X, Y)^T$ – координаты точки в системе координат доски, K – матрица внутренних параметров камеры, $R = (r_1\ r_2\ r_3)$ – матрица поворота, t – вектор перемещения.

С помощью найденной матрицы H определяются координаты всех точек пересечений клеток доски на изображении. Для уточнения координат ключевых точек используется субпиксельное уточнение.

Рис. 1: Примеры детекции ключевых точек доски

После нахождения ключевых точек на первом кадре происходит их отслеживание алгоритмом Лукаса – Канаде [1], в основе которого лежит идея оптического потока.

2.2. Определение положения камеры

Оценка положения камеры осуществляется известными методами по парам соответствующих трехмерных точек и их проекций на плоскость камеры.

Координаты точек в пространстве и на изображении связаны уже ранее упоминающимся уравнением проектирования в однородных координатах:

$$s \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K(R\ t) \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix},$$

где $(X, Y, Z)^T$ – координаты точки в системе координат доски, дополненной до трехмерной путем добавления оси, перпендикулярной плоскости доски.

На основе известных координат ключевых точек составляется система уравнений относительно неизвестных параметров R и t. Для ее решения необходимо знать матрицу внутренних параметров камеры, поэтому перед проведением экспериментов камера калибруется с помощью Camera Calibration Toolbox for Matlab [2]. Используя данные, полученные в результате калибровки, составляется матрица K и исправляется дисторсия на изображениях, получаемых с камеры.

Решение полученной системы производится методом Левенберга – Марквардта для решения задач нелинейных наименьших квадратов, реализованным в библиотеке LMFIT [7]. Для работы данного метода необходимо первоначальное приближение

решения, в качестве которого берется матрица гомографии, полученная на этапе детекции точек. Но матрица $H = K(r_1\ r_2\ t)$, размерности 3×3, не соответствует решению вида $(R\ t) = (r_1\ r_2\ r_3\ t)$, размерности 3×4. В связи с этим производятся следующие преобразования H:

- Домножение на K^{-1} слева
- Ортогонализация и нормирование векторов r_1 и r_2 методом Грама – Шмидта
- Нахождение вектора r_3, как векторного произведения r_1 и r_2

Таким образом, решая для каждого кадра получаемую систему и находя матрицу поворота и вектор перемещения, производится оценка положения камеры относительно неподвижного маркера.

2.3. Детекция движущегося объекта

Для детекции движущихся точек используется алгоритм на основе отслеживания особенных точек. Необходимо определить и отследить точки, определить, что точки принадлежат одному и тому же объекту. Детекция точек производится детектором Shi-Tomasi. Отслеживание производится методом Лукаса – Канаде. Оба метода реализованы в библиотеке OpenCV [8]. Для каждой точки рассматривается отдельный объект, затем эти объекты могут быть собраны в супер-объект.

2.4. Алгоритм определения глубины движущейся точки

Для определения положения движущегося объекта достаточно оценивать глубину, так как положение может быть восстановлено по формуле:

$$P = D\frac{R^{-1}(x, y, 1)^T}{\|R^{-1}(x, y, 1)^T\|},$$

где $P = (X, Y, Z)^T$, D - положение и глубина точки, (x, y) - координаты точки на изображении, R - ориентация камеры.

Рассматривается динамическая система с состоянием $\theta_n \in \mathbf{R}^3$, где $n \in \mathbf{N}$ – дискретный момент времени. $(\theta_n^{(1)}, \theta_n^{(2)})$ – координаты точки на изображении, $\theta_n^{(3)}$ – обратная глубина точки.

Система координат определяется камерой. Предполагается, что точка свободно двигается в пространстве, $\xi_n = (\xi_n^{(1)}, \xi_n^{(2)}, \xi_n^{(3)})^T$ – вектор перемещения точки. В то же время камера в каждый момент, не меняя ориентацию, совершает движение $\Delta_n = (\Delta_n^{(1)}, \Delta_n^{(2)}, \Delta_n^{(3)})^T$.

Предлагается следующая последовательность действий:

- Камера совершает случайное перемещение Δ_n

- Производятся измерения y_n
- Обновляются оценки истинной проекции точки на камеру и ее обратной глубины

Основной алгоритм имеет вид

$$\hat{\theta}_{n+1} = \hat{\theta}_n - \alpha(\hat{\theta}_n - y_n)$$

где $\alpha > 0$ – коэффициент размера шага.

Предлагается делать наблюдения следующим образом:

$$y_n^{(1)} = \frac{X_{n-1} + \Delta_n^{(1)} + \xi_n^{(1)}}{Z_{n-1} + \Delta_n^{(3)} + \xi_n^{(3)}} + v_n^{(1)}$$

$$y_n^{(2)} = \frac{Y_{n-1} + \Delta_n^{(2)} + \xi_n^{(2)}}{Z_{n-1} + \Delta_n^{(3)} + \xi_n^{(3)}} + v_n^{(2)}$$

$$y_n^{(3)} = k_n \Delta_n^{(1)} \left(\frac{X_{n-1} + \Delta_n^{(1)}}{Z_{n-1} + \Delta_n^{(3)} + \xi_n^{(3)}} - \frac{X_{n-1}}{Z_{n-1}} + v_n^{(1)} - v_{n-1}^{(1)} \right)$$

где $k_n = |\Delta_n^{(1)}|^{-2}$, (X_n, Y_n, Z_n) – координаты точки в пространстве, $(v_n^{(1)}, v_n^{(2)})$ – неизвестный вектор шума.

3. Результаты экспериментов

Эксперименты проводились при помощи роботов "ТРИК". Все описанные выше алгоритмы были реализованы на языке Python. Для тестирования алгоритмов записывался видеопоток с камер, и дальнейшая работа производилась с полученными видео.

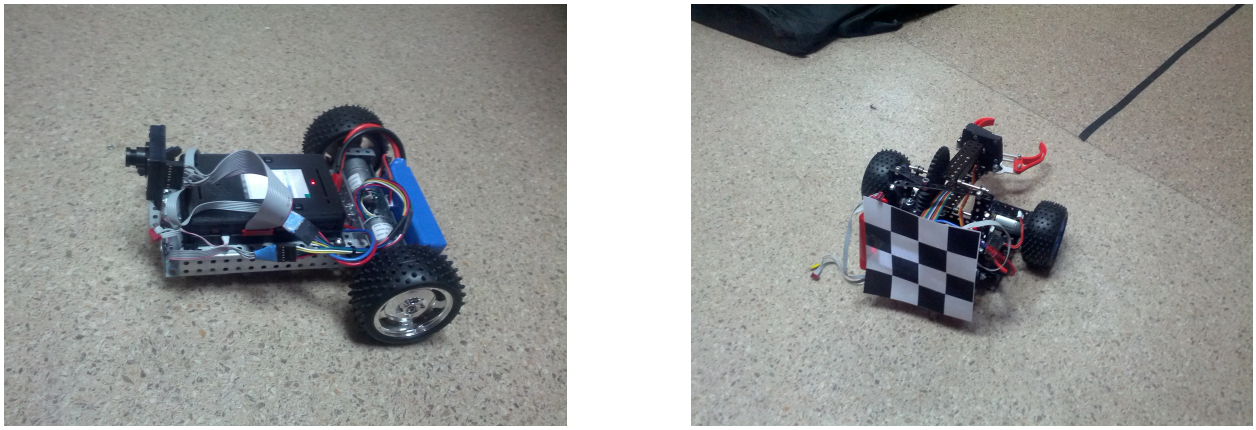

Рис. 2: Роботы "ТРИК", используемые в экспериментах

3.1. Определение положения камеры

Прежде чем применять составленный алгоритм определения положения камеры на практике, он был протестирован на сегенрированных данных. На рисунках 3 и 4 представлены графики ошибок определения расстояния до неподвижного маркера при случайном движении камеры в небольшой окрестности своего начального положения. Для симуляции реальных условий к измерениям координат ключевых точек были добавлены случайные помехи. В результате тестирования был сделан вывод, что средняя ошибка работы алгоритма на 100 итерациях состовляет 2-4%, что позволяет дальнейшее рассмотрение данного алгоритма.

Затем были проведены эксперименты с роботом по следующей схеме: робот двигался по прямой по направлению к неподвижному маркеру, начиная с заранее измеренного расстояния, по ходу своего движения робот останавливался на отмеченных линиях, расстояния до которых также были измерены, в эти моменты определялось положение и сравнивалось с истинным значением. Усредненные по 10 экспериментам значения ошибок на каждой линии представлены в таблице 1.

Помимо этого для оценки работы алгоритма был применен косвенный метод на базе доски. В результате применения метода Левенберга – Марквардта находилась матрица ковариации неизвестных параметров. Корень из диагонального элемента, соответствующего одному из параметров, является оценкой стандартного отклонения этого параметра. Данный метод оценивания описан в книге [3] и является стандарт-

ным подходом в фотограмметрии. График стандартного отклонения оцениваемого расстояния до маркера в одном из проведенных экспериментов представлен на рисунке 5.

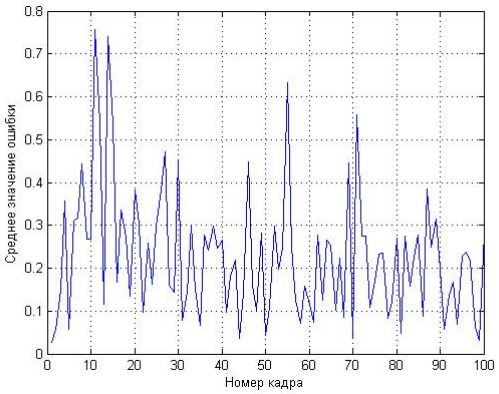

Рис. 3: Начальное положение $(0, 0, 10)$

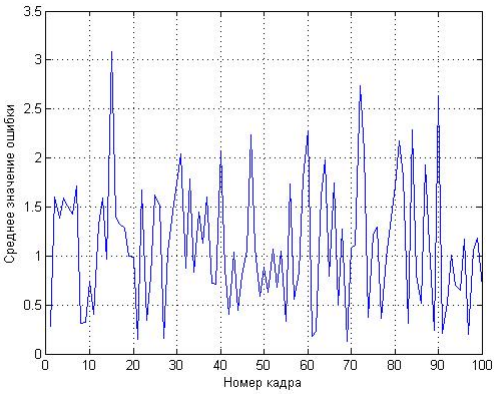

Рис. 4: Начальное положение $(0, 0, 30)$

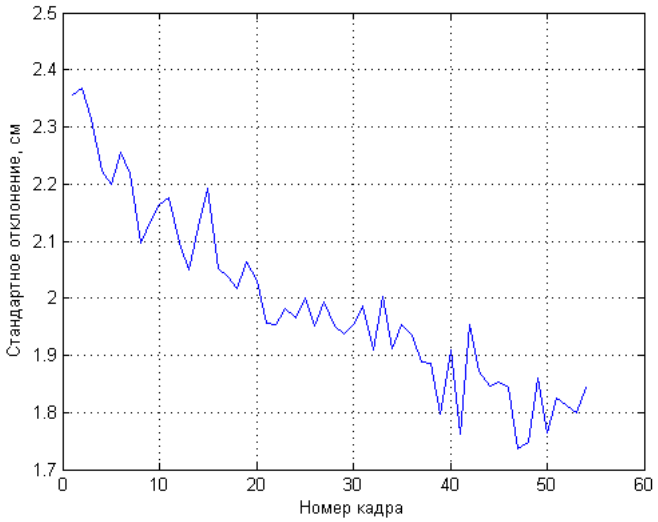

Рис. 5: График оценки стандартного отклонения

Расстояние, см	Оценка, см	Ошибка, см	Ошибка, %
60	57.815	2.185	3.642
90	85.253	4.747	5.274
120	116.411	3.589	2.991
150	146.609	3.391	2.261
180	174.519	5.481	3.045
210	204.229	5.771	2.748
240	234.517	5.483	2.284

Таблица 1: Значения ошибок определения расстояния до неподвижного маркера

3.2. Алгоритм определения глубины движущейся точки

Чтобы протестировать алгоритм определения дальности [4] были проведены эксперименты с двумя движущимися роботами. Для детекции и отслеживания на одном из роботов был прикреплен маркер (см. рис. 2), а для определения каждым роботом своего положения были установлены два неподвижных маркера (см. рис. 6) с известным расстоянием между ними.

Эксперименты проводились следующим образом. Первый робот управлялся вручную с помощью специального приложения. В различных экспериментах он двигался либо от маркеров к другому роботу, либо наоборот. Второй двигался вперед-назад в окрестности своего начального положения, симулируя случайные движения, необходимые для работы алгоритма. Каждый робот должен был видеть один из неподвижных маркеров, при этом второй робот должен был видеть еще маркер, установленный на первом. Схема проведения одного из экспериментов, полученная после применения алгоритма определения положения камеры к видео с камеры каждого робота, представлена на рисунке 7.

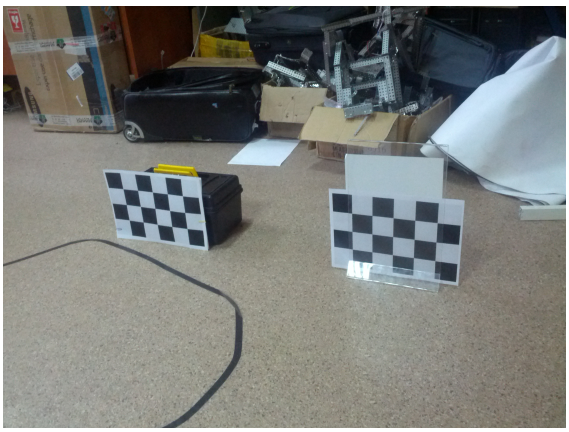

Рис. 6: Расположение неподвижных маркеров

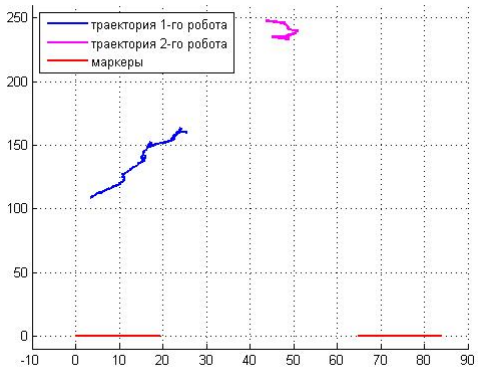

Рис. 7: Схема проведения эксперимента

Для анализа получаемых в ходе работы алгоритма данных измерялось изменение оцениваемой глубины между последовательными кадрами и изменение глубины, получаемой исходя из данных определения положения каждого робота. График получаемых ошибок в одном из экспериментов представлен на рисунке 8, на котором видно, что ошибки получились довольно большими, порядка 100-150%, но на этом сказались погрешнось алгоритма определения положения камеры и возможное расхождение данных во времени.

После этого стало понятно, что необходима более точная оценка алгоритма. В связи с этим провелись эксперименты, когда отслеживаемый робот останавливался на линиях с заранее известным расстоянием до них, и истинная глубина точки сравнивалась с оцениваемой.

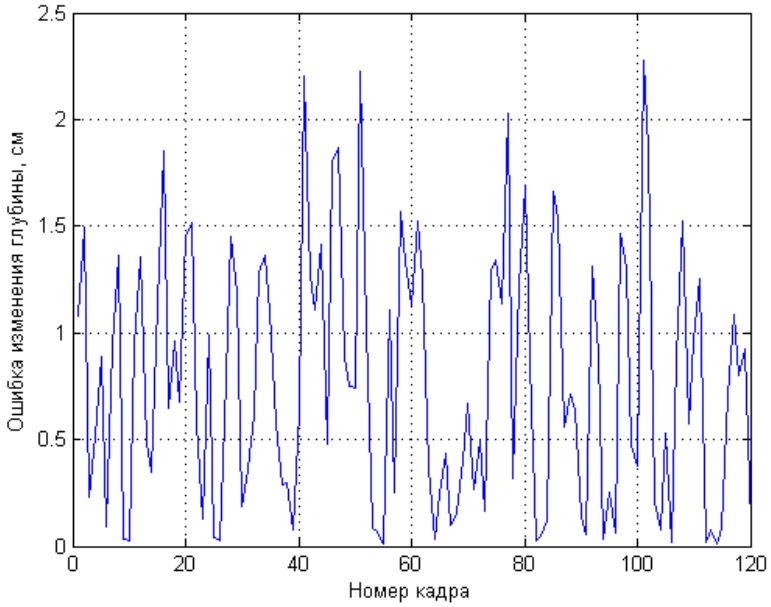

Рис. 8: График ошибки изменения глубины

Усредненные по 10 экспериментам значения ошибок на каждой линии представлены в таблице 2.

Глубина, см	Оценка, см	Ошибка, см	Ошибка, %
80	92.635	12.635	15.794
110	136.965	26.965	24.514
140	113.989	26.011	18.579
170	146.409	23.591	13.877
200	239.328	39.328	19.664

Таблица 2: Значения ошибок оценки глубины движущейся точки

Заключение

В ходе работы были выполнены следующие задачи:

- Составлен и реализован алгоритм определения положения камеры
- Реализован алгоритм оценки глубины движущейся точки
- Проведены эксперименты с роботами для исследования реализованных алгоритмов
- Удалось экспериментально подтвердить работоспособность алгоритмов

В дальнейшем планируется перейти от определения собственного положения по шахматной доске к решению задачи Structure from motion в общем виде, а также провести ряд модификаций алгоритма определения глубины движущейся точки, например, применение фильтра Калмана.

Список литературы

[1] Bradski Gary, Kaehler Adrian. Learning OpenCV: Computer vision with the OpenCV library. — " O'Reilly Media, Inc.", 2008.

[2] Camera Calibration Toolbox for Matlab. — URL: `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html` (online; accessed: 20.05.2015).

[3] Hartley Richard, Zisserman Andrew. Multiple view geometry in computer vision. — Cambridge university press, 2003.

[4] Krivokon Dmitry, Vakhitov Alexander. Randomized Algorithm for Estimation of Moving Point Position Using Single Camera // Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on. — 2014. — P. 5189–5194.

[5] Markoff John. Google Cars Drive Themselves, in Traffic // New York Times. — 2010. — Vol. 9.

[6] MonoSLAM: Real-time single camera SLAM / Andrew J Davison, Ian D Reid, Nicholas D Molton, Olivier Stasse // Pattern Analysis and Machine Intelligence, IEEE Transactions on. — 2007. — Vol. 29, no. 6. — P. 1052–1067.

[7] Non-Linear Least-Square Minimization and Curve-Fitting for Python. — URL: `http://lmfit.github.io/lmfit-py/index.html` (online; accessed: 20.05.2015).

[8] Open source computer vision. — URL: `http://opencv.org/` (online; accessed: 20.05.2015).

[9] Papanikolopoulos Nikolaos P, Khosla Pradeep K, Kanade Takeo. Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision // Robotics and Automation, IEEE Transactions on. — 1993. — Vol. 9, no. 1. — P. 14–35.

[10] Weiner Tim. A new model army soldier rolls closer to the battlefield // New York Times. — 2005. — Vol. 16. — P. A1.

[11] Yamaguchi Koichiro, Kato Takeo, Ninomiya Yoshiki. Vehicle ego-motion estimation and moving object detection using a monocular camera // Pattern Recognition, 2006. ICPR 2006. 18th International Conference on / IEEE. — Vol. 4. — 2006. — P. 610–613.