

Нгуен Туан Зунг, аспирант, dunghvkt@yahoo.com,
И. А. Щербатов, канд. техн. наук, доц., Sherbatov2004@mail.ru,
Астраханский государственный технический университет, г. Астрахань

Совместное распознавание подвижных и неподвижных объектов в системе технического зрения робота

Предложены модифицированные алгоритмы совместного обнаружения подвижных и неподвижных объектов, которые разделяются на два типа: известные объекты (неподвижные препятствия, например, стол, стул, компьютер и пр.) и подвижные объекты (например, люди). Представлены модифицированные алгоритмы распознавания для каждого типа объектов: для неподвижных объектов применяется модифицированный алгоритм поиска ближайших соседей совместно с деревьями поиска (KNN) и библиотекой FLANN; для подвижных объектов типа "человек" применены встроенные алгоритмы комплекта разработчика (SDK) Microsoft Kinect. Показана эффективность алгоритма поиска ближайших соседей при обнаружении неподвижных объектов. Эффективность нахождения объектов увеличена за счет применения алгоритма SURF. Результаты экспериментов показывают эффективность предложенного подхода при использовании в составе системы технического зрения мобильной робототехнической платформы.

Ключевые слова: FLANN, SURF, мобильный робот, распознавание изображений, поиск ближайшего соседа, изображения глубины, предварительная обработка глубинных данных, Kinect SDK

Введение

Для автономной навигации мобильных роботизированных платформ в зданиях и сооружениях необходимо решить задачу объезда подвижных (человек) и неподвижных (предметы, мебель, оборудование и пр.) препятствий. Для этого система технического зрения робота должна обнаруживать и распознавать максимально возможное число препятствий, обеспечивая перемещение без столкновений.

Обнаружение не единственного объекта на фотографиях или в потоке видеоизображений является сложной задачей. В последнее время проведено значительное число исследований в области обнаружения человека или неподвижных объектов, а также предложены различные методы их распознавания. Большинство исследований базируются на изображениях, полученных из видимого спектра, по аналогии с тем, как это воспринимается человеческим глазом. Некоторые методы используют статистические выборки на основе локальных особенностей (например, HOG — Histograms of Oriented Gradients, EOH — Edge Orientation Histograms [1]), а также методы извлечения особых точек изображения (например, SIFT — Scale Invariant Feature Transform [2], SURF — Speeded Up Robust Features [3]).

Данные методы позволяют получить достаточно точные результаты по обнаружению объектов, но являются критичными по отношению к распознаванию человека или изменению фона изображения. Это приводит к снижению точности распознавания и увеличению вычислительных мощностей, а значит, и времени распознавания объекта [4].

Для устранения указанных недостатков и распознавания объектов могут применяться данные глубин изображений, различающихся по цвету и текстуре и занимающие определенную область в пространстве. Карты глубин имеют ряд преимуществ

перед двухмерными изображениями, так как являются устойчивыми к изменению цвета и освещения и простыми с точки зрения отображения [5]. Дороговизна сенсоров глубины (в которых применялись лазеры) долгое время препятствовала активному их внедрению. Однако с появлением Microsoft Kinect существенным образом может быть повышено качество и скорость распознавания подвижных препятствий (в том числе людей) при автономной навигации роботов.

Постановка задачи

На обнаружение объектов влияет ряд факторов: масштаб, место, фон и помехи, проекция, вращение и угол обзора. Для снижения их влияния необходимо разработать алгоритм обнаружения, использующий информацию о глубинах изображений, получаемых с использованием сенсоров Kinect. Для этого требуется решить две взаимосвязанные подзадачи: обнаружение подвижных объектов (людей) с применением карты глубин изображений; распознавание неподвижных объектов с использованием метода поиска ближайшего соседа.

Обозначим множество распознаваемых объектов в ближней зоне мобильного робота: $O = \{O_p, O_n\}$, где O_p — множество подвижных объектов, O_n — множество неподвижных объектов. Каждый объект $O_{p,i}$ ($i = 1, \dots, i_p$, i_p — число подвижных объектов) и $O_{n,j}$ ($j = 1, \dots, j_n$, j_n — число неподвижных объектов) описывается тройкой: $O_{p,i} = \{T_{p,i}, \Theta_{p,i}, \Phi_{p,i}\}$, $O_{n,j} = \{T_{n,j}, \Theta_{n,j}, \Phi_{n,j}\}$, где T — тип фигуры, Θ — геометрические размеры фигуры, Φ — угол обзора.

Для распознавания используются эталонные объекты $O_{э,k}$ ($k = 1, \dots, k_э$, $k_э$ — число эталонных объектов), содержащиеся в базе данных системы распознавания. Тогда задача распознавания объекта в ближней зоне робота формулируется следующим образом: необходимо присвоить всем распознаваемым

мым объектам признак $\alpha = 1$, если объект распознан, и $\alpha = 0$ — в противном случае, т. е.

$$\forall O_l \in O \quad \alpha = \begin{cases} 1, & \text{если } |f(O_l) - f(O_3)| \leq \varepsilon; \\ 0, & \text{если } |f(O_l) - f(O_3)| > \varepsilon, \end{cases} \quad (1)$$

где $f(O_l)$ — числовой показатель, характеризующий дескриптор изображения распознаваемого объекта; $f(O_3)$ — числовой показатель, характеризующий дескриптор эталонного изображения; $\alpha = 1$ — признак того, что объект распознан; $\alpha = 0$ — признак того, что объект не распознан (отсутствует в базе эталонных изображений объектов); ε — числовая граница точности распознавания объекта.

Решение задачи в такой постановке требует синтеза скоростной процедуры распознавания объектов, позволяющей обеспечить требуемую точность распознавания $\Gamma = |\Gamma' - \Gamma''|$, сократив при этом время, затрачиваемое на одновременную идентификацию подвижных $\tau_{\text{п}}$ и неподвижных объектов $\tau_{\text{н}}$ в ближней зоне работы, где $\Gamma' = f(O_l)$, $\Gamma'' = f(O_3)$.

Общий алгоритм распознавания объектов

При отсутствии света использование камер затруднительно. В последнее время в данной области произошел существенный сдвиг в связи с появлением инфракрасных сенсоров глубины, таких как Kinect [6]. Датчик глубины Kinect состоит из инфракрасного проектора, объединенного с монохромной КМОП-матрицей, что позволяет ему получать трехмерное изображение при любом естественном освещении [5]. Основа работы системы распознавания объектов — алгоритм, показанный на рис. 1. Опишем каждый из блоков.

Данные глубины. Пиксели на изображении глубины указаны калиброванными глубинами пространства перед роботом и не являются мерой интенсивностей или цветов. Получаемые изображения имеют разрешение 640×480 пикселей при 30 кадрах в секунду, разрешение по глубине — несколько сантиметров. Камеры глубины имеют ряд преимуществ перед традиционными датчиками интенсивности, работающими при низком уровне освещенности, позволяя получать калиброванную масштабную оценку, инвариантную по отношению к цвету и текстуре. Kinect также значительно упрощает решение задачи "вычитания" фона, которая используется в синтезированном алгоритме. Ключевым аспектом выполнения данного блока является синтез реалистичных глубин изображения подвижных объектов (в том числе людей).

Предварительная обработка данных глубины. Первый шаг — захват RGB-

изображения и соответствующих ему глубин. После этого выполняется предварительная обработка данных глубины, а также выделение объекта.

Kinect предназначен для измерения расстояния, начиная от 60 см до нескольких метров. Любые препятствия, расположенные вне этого диапазона, имеют нулевые значения глубин. Эти области будут давать ошибочные результаты, поэтому требуют коррекции. Для этих целей применяется ускоренный алгоритм, обрабатывающий полученные значения глубин и заменяющий нулевые значения ближайшими ненулевыми (рис. 2).

Процесс нормализации изображения обеспечивается следующей последовательностью шагов: запоминание данных о глубине в виде матрицы; поиск нулевых значений глубин; все нулевые значения глубин считаются кандидатами на фильтрацию; фильтр создает частотное распределение значений глубин, учитывая найденные глубины в каждом диапазоне для определения кандидата, который должен быть отфильтрован; если пороговое значение для каждого диапазона нарушается, то статистический режим всех значений ненулевых глубин будет применен к кандидатам, в противном случае никаких замен не проводится.

Обнаружение подвижного объекта. После подключения к Kinect происходит получение данных цвета, глубины и скелета потоков данных ("skeletondata") [4]. Блок "Предварительная обработка данных глубины" обеспечивает удаление нулевых значений глубин из массива (рис. 3). Далее обработчик событий останавливается в целях ожидания

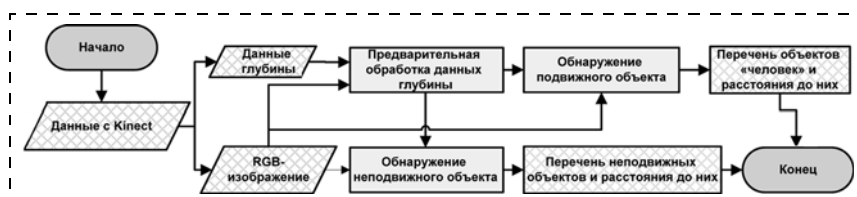


Рис. 1. Общий алгоритм распознавания объектов

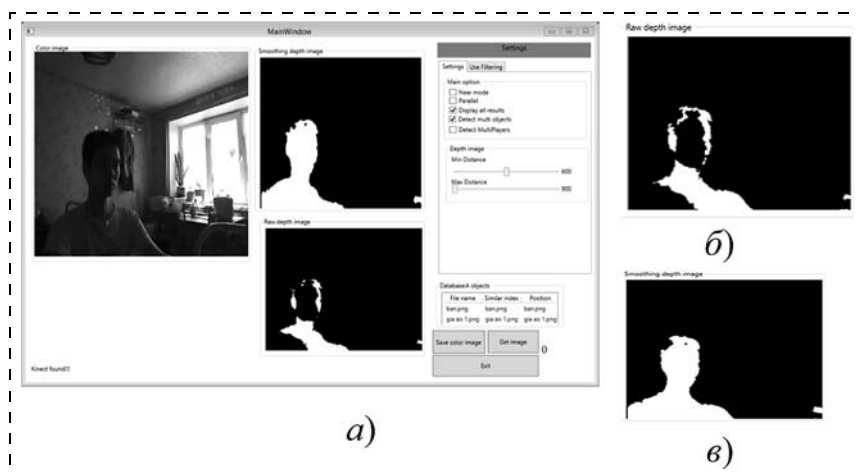


Рис. 2. Получение глубин изображения с датчика Kinect:

а — интерфейс модуля; б — входное изображение; в — нормализованное изображение

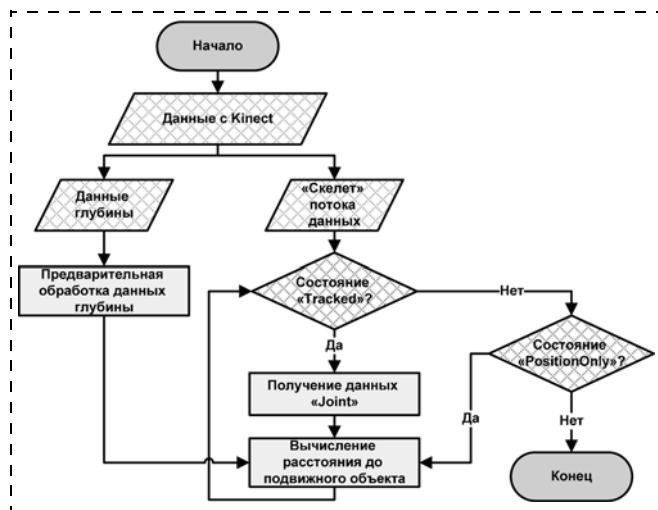


Рис. 3. Алгоритм обнаружения подвижного объекта

получения готовности всех кадров с Kinect. Наличие готовности означает, что получены данные глубин кадров, кадр цветного изображения и "скелет" кадра одновременно. Указанные данные хранятся в массивах глубины изображения, цветного изображения и точек каркаса.

Для получения местоположения людей в кадре используется информация, хранящаяся в массиве глубин [5]. Каждый пиксель в этом массиве кодирует всю информацию о глубине и объекте. При использовании битовой маски к этой части данных можно извлечь информацию об объекте "человек". Алгоритм, который Microsoft внедряет в свой комплект разработчика (SDK), возвращает местоположение человека в 200 кадров/с [4].

При использовании Kinect любые отражающие поверхности идентифицируются значениями глубины. Этот сигнал является простым способом выделения "плохих" данных. Для выдачи масштабированных данных исходные данные глубины возвращаются в виде некоторого числа пикселей и связанных с ними значений глубины в миллиметрах. Проблема преобразования пикселей в миллиметры решается с использованием функций комплекта SDK.

Ориентация конечностей (существенных признаков подвижного объекта "человек") в Kinect представляется в двух форматах: иерархическое вращение в точке, соответствующей суставу человека; абсолютная ориентация в координатах камеры Kinect. Информация об ориентации представляется в виде кватернионов и вращений матриц.

У объекта существует три состояния: Tracked (человек может быть отслежен в деталях), PositionOnly (местоположение человека получено), NotTracked (человек не может быть отслежен). Полученные данные называются Joint и имеют информацию о главных координатах человека (например, координатах центра головы, центра правой руки, центра левой руки и т. д.). Если присутствует состояние

PositionOnly, то координаты центра подвижного объекта (человека) получены.

Обнаружение неподвижных объектов. Для распознавания неподвижных объектов применяются два алгоритма: линейный поиск (требует существенных вычислительных затрат) [7] и приближенный поиск ближайших соседей (обеспечивает поиск с требуемой точностью за приемлемое время) [8].

В алгоритме поиска ближайших соседей (алгоритм NNS) существует задача оптимизации нахождения ближайших точек. Близость обычно выражается в терминах функции несходства. Однако наиболее часто применяется приближенный поиск ближайших соседей, в котором существуют три известных алгоритма: FLANN (Fast Library for Approximate Nearest Neighbors) [8], ANN (A Library for Approximate Nearest Neighbor Searching) [9], STANN (Thread-safe Approximate Nearest Neighbor) [10]. В робототехнике наиболее часто применяется алгоритм FLANN, поскольку он поддерживается в большинстве библиотек, таких как OPENCV и OPENNI [11].

В алгоритме поиска ближайших соседей применяются следующие правила расчета расстояния d между двумя точками (x и y):

$$\begin{cases} d(x, y) \geq 0, d(x, y) = 0, \text{ если } x = y; \\ d(x, y) = d(y, x); \\ d(x, z) \leq d(x, y) + d(y, z), \text{ если точки } x, y, z \\ \text{не лежат на одной прямой.} \end{cases}$$

Евклидово расстояние находится для упорядоченных значений атрибутов (перед расчетом расстояния необходима нормализация [12]):

$$D_E = \sqrt{\sum_i^n (x_i - y_i)^2},$$

где n — число атрибутов.

Функция несходства применяется для строковых переменных, которые не могут быть упорядочены. Она задается следующим образом:

$$d(x, y) = \begin{cases} 0, x = y; \\ 1, x \neq y. \end{cases}$$

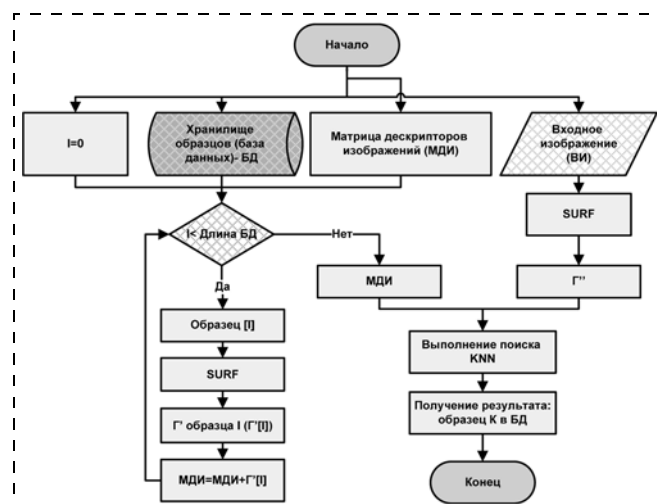


Рис. 4. Схема применения библиотеки FLANN

Поиск ближайшей точки является алгоритмом с асимптотической сложностью $O(\log N)$ в случае хаотически распределенных точек. Данный алгоритм совместно с алгоритмом аппроксимации может работать быстрее. Для решения задачи NNS используется библиотека FLANN (рис. 4). Эта библиотека содержит коллекцию следующих дескрипторов для множества изображений: создание индекса FLANN; выполнение поиска KNN; получение и вывод результатов. Для ускорения вычислений применяются следующие алгоритмы: SURF для пошагового "извлечения дескрипторов изображения"; параллельное программирование.

Эксперимент и анализ результатов

Программная реализация рассматриваемых алгоритмов была выполнена с применением библиотеки FLANN в среде разработки Visual Studio C# 2010. Эксперименты проводили для одного, двух и трех неподвижных объектов (рис. 5) на персональной ЭВМ со следующими характеристиками: центральный процессор Intel Pentium Duo T2390 с частотой 1,87 ГГц, объем оперативной памяти 1 Гб, объем видеопамати 256 Мб. Полученные результаты представлены на рис. 6.

Из табл. 1 видно, что разработанная программа успешно обнаруживает неподвижные объекты. Присутствие на рис. 6 скачков времени обусловлено тем, что алгоритм использует изображения, полученные в видимом спектре (изменяются условия освещения для различных кадров), а также из-за движения объектов по отношению к Kinect (движение мобильной платформы робота, на котором он установлен). Оба эти аспекта оказывают влияние на скорость (время) распознавания объекта.

Для сокращения среднего времени обнаружения неподвижного объекта и повышения быстродействия разработанного алгоритма применено параллельное программирование (результаты показаны на рис. 7). Наличие скачков обусловлено описанными выше причинами.

Для распараллеливания распознавания нескольких объектов применен алгоритм, входящий в состав библиотеки AForge, которая позволяет задействовать все ядра микропроцессора (по одному потоку на ядро). Таким образом, каждый объект распознается на отдельном ядре микропроцессора. В табл. 2 представлены численные значения среднего времени обнаружения объектов при использовании традиционного подхода и распараллеливания распознавания объектов (обеспечивается повышение скорости обнаружения более чем на 3 %).

Таблица 1

Обнаружение нескольких неподвижных объектов (сравнение результатов)

Число обнаруживаемых объектов	Число входных изображений	Ошибка, шт. (%)	Среднее время обнаружения объекта, мс
1	43	4 (≈ 9)	≈ 403
2	43	7 (≈ 16)	≈ 481
3	43	7 (≈ 16)	≈ 486

Таблица 2

Сравнение традиционного и параллельного программирования

№	Вид эксперимента	Число входных изображений	Среднее время обнаружения объекта, мс
1	Без применения параллельного программирования	43	402,77
2	С применением параллельного программирования	43	389,84

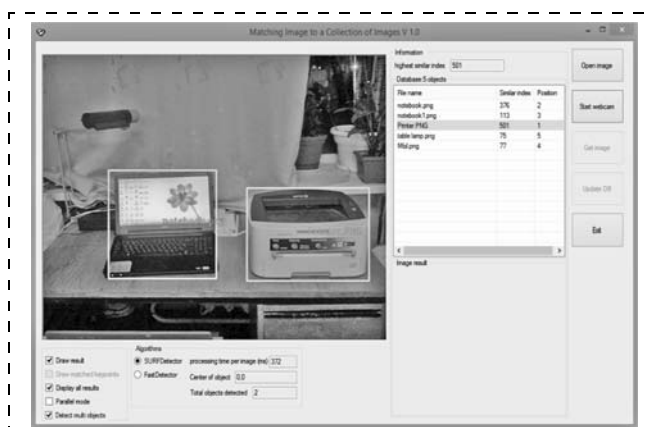


Рис. 5. Интерфейс программы распознавания неподвижных объектов

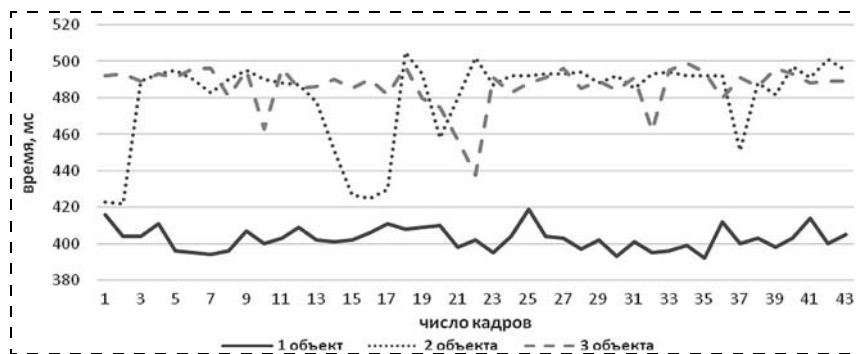


Рис. 6. Быстродействие алгоритма распознавания неподвижных объектов



Рис. 7. Сравнение результатов при традиционной реализации и при использовании параллельного программирования



Рис. 8. Интерфейс программы совместного распознавания подвижных и неподвижных объектов

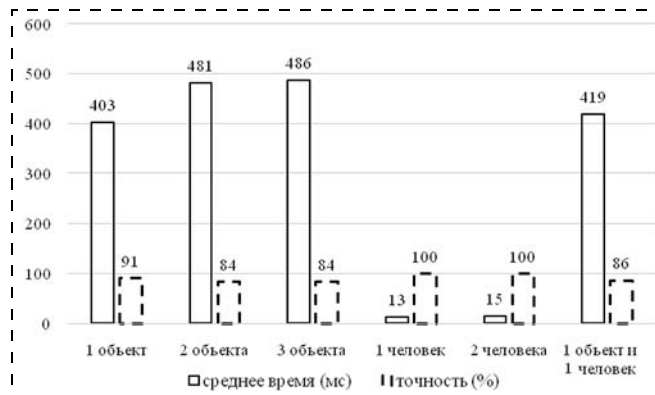


Рис. 11. Сводная диаграмма результатов исследования

Таблица 3
Сравнительный анализ результатов

№	Объекты	Число входных изображений	Ошибка, шт. (%)	Среднее время обнаружения объекта, мс
1	1 человек	43	0 (= 0 %)	13,23
2	2 человека	43	0 (= 0 %)	14,65
3	2 объекта	43	7 ($\approx 16,27$ %)	480,88
4	1 объект и 1 человек	43	2 ($\approx 4,65$ %)	419,11

Вторая серия экспериментов была направлена на реализацию алгоритмов, обеспечивающих совместное распознавание подвижных (например, человек) и неподвижных объектов (например, стол, принтер и пр.), интерфейс программы представлен на рис. 8. Результаты экспериментов показаны на рис. 9, 10 и приведены в табл. 3 (расстояние до подвижного объекта "человек" — 1,9 м, а до неподвижного объекта "принтер" — 2,58 м).

Из табл. 3 видно, что при использовании Kinect обеспечивается хорошее качество распознавания подвижных объектов. Результаты экспериментов свидетельствуют о том, что существует возможность совместного распознавания нескольких подвижных и неподвижных объектов в режиме реального времени с применением Kinect. Однако для увеличения быстродействия традиционных алгоритмов распознавания следует применять параллельное программирование. Для удобства сравнения полученных в рамках исследования результатов все полученные данные сведены к одной диаграмме, демонстрирующей эффективность предложенных в работе алгоритмов (рис. 11).

Заключение

Данные экспериментов показывают, что обнаружение подвижных и неподвижных объектов следует проводить с использованием различных алгоритмов. Для распознавания неподвижных объектов целесообразно применять алгоритм поиска ближайших соседей, модифицированного для применения совместно с деревьями поиска (KNN) и библиотекой FLANN. Для распознавания подвижных объектов типа "человек" следует использовать встроенные алгоритмы комплекта



Рис. 9. Быстродействие алгоритма распознавания подвижных объектов

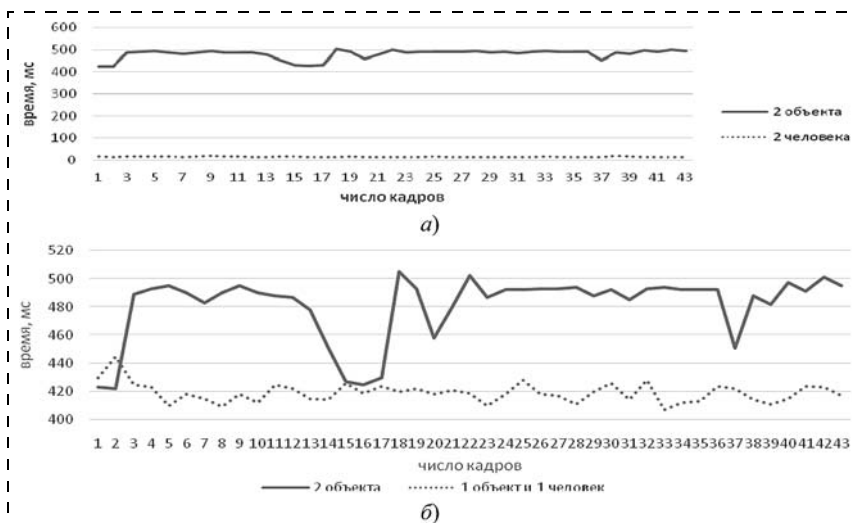


Рис. 10. Быстродействие алгоритма совместного распознавания подвижных и неподвижных объектов:
а — сравнение при распознавании двух подвижных и двух неподвижных объектов;
б — сравнение при распознавании двух неподвижных объектов и совместного распознавания одного неподвижного и одного подвижного объекта

разработчика (SDK) Microsoft Kinect. При этом применение распараллеливания программной обработки поиска нескольких подвижных и неподвижных объектов существенно увеличивает быстроедействие алгоритма совместного распознавания объектов и повышает его точность.

Полученные результаты свидетельствуют о том, что использование библиотеки FLANN совместно с Kinect SDK удовлетворяет жестким требованиям по обнаружению объектов в режиме реального времени. Это означает, что предложенный подход может применяться в системах технического зрения мобильных робототехнических платформ для получения сенсорной карты внешней среды, используемой системой управления роботом для перемещения в зданиях и сооружениях в заранее неизвестной среде.

Список литературы

1. Navneet Dalai, Bill Triggs. Object Detection using Histograms of Oriented Gradients. URL: <http://www.webcitation.org/6DvoEuAvL> (дата обращения: 14.10.2014).
2. Ramisa A., Vasudevan S., Aldavert D., Toledo R., Lopez de Mantaras R. Evaluation of the SIFT Object Recognition Method in Mo-

bile Robots. URL: http://www.iiia.csic.es/~mantaras/sift_eval.pdf (дата обращения: 14.10.2014).

3. Bay H., Ess A., Tuytelaars T., Van Gool L. Speeded-Up Robust Features (SURF) // Computer Vision and Image Understanding. 2008. Vol. 110, Issue 3. P. 346—359.
4. Marron R., Janet J. C# Implementation of SLAM Using the Microsoft Kinect. URL: http://old.ee.duke.edu/files/ece/GWD-D2012_Marron.pdf (дата обращения: 14.10.2014).
5. Shotton J., Fitzgibbon A. Cook M., Sharp T., Finocchio M., Moore R., Kipman A., Blake A. Real-time human pose recognition in parts from single depth images // VISAPP International Conference on Computer Vision Theory and Applications. February 2013. Pp. 119—135.
6. Kefer M., Kubinger W. Evaluation of Kinect Depth Sensor for Use in Mobile Robotics // Annals of DAAAM for 2011 & Proceedings of the 22nd International DAAAM Symposium. 2011. Vol. 22, N. 1.
7. Knuth D. Section 6.1: Sequential Searching. Sorting and Searching. The Art of Computer Programming 3. Addison-Wesley. 1997. P. 396—408.
8. Muja M., Lowe D. G. Fast approximate nearest neighbors with automatic algorithm configuration // VISAPP International Conference on Computer Vision Theory and Applications. 2009. P. 331—340.
9. Mount D. M., Arya S. ANN: A Library for Approximate Nearest Neighbor Searching. URL: <http://www.cs.umd.edu/~mount/ANN/> (дата обращения: 05.11.2014).
10. Connor M. The Simple, Thread-safe Approximate Nearest Neighbor (STANN). URL: <https://sites.google.com/a/compgeom.com/stann/Home> (дата обращения: 05.11.2014).
11. Forsyth D. A., Ponce J. Computer Vision: A Modern Approach // Robotics and Autonomous Systems. 2006. Vol. 11, N. 3. P. 84—92.
12. Обзор информации об алгоритме ближайшего соседа. URL: <http://www.basegroup.ru/library/analysis/regression/knn/> (дата обращения: 05.11.2014).

Joint Recognition of the Moving and Stationary Objects in the Machine Vision Systems of Robots

Nguyen Tuan Dung, dunghvkt@yahoo.com, I. A. Shcherbatov, shcherbatov2004@mail.ru, Astrakhan, 414056, Russian Federation

Corresponding author: Shcherbatov Ivan A., Ph. D., Associate Professor, Astrakhan State Technical University, Astrakhan, 414056, Russian Federation e-mail: shcherbatov2004@mail.ru

Received on December 18, 2014

Accepted December 28, 2014

Simultaneous detection of multiple stationary and moving obstacles in the near field of the mobile robots is a challenging task, since a robot has to detect a maximal possible number of obstacles, and ensure its movement without collisions. In this paper, the authors propose modified algorithms for detection of objects. Detectable objects are divided into two types: familiar objects (stationary obstacles, for example, a table, a chair, a computer, etc.), and unknown objects (moveable objects — people).

The authors present specific recognition algorithms for each object type: the nearest neighbor search algorithm modified for the use with FLANN library and search trees (KNN) used for detection of the stationary obstacles; the built-in algorithms (Microsoft Kinect development kit-SDK) are intended for recognition of such moveable objects as persons. The efficiency of the search algorithm of the nearest neighbors for detection of stationary objects is shown. This algorithm is implemented in FLANN library, which contains main algorithms for extraction of the handles of images and creation of indexes. The effectiveness of finding objects is increased due to application of SURF algorithm. Use of FLANN Library together with SURF algorithm satisfies the requirements for detection of objects in real time. The experimental results prove the effectiveness of the proposed approach in a vision system of a mobile robotic platform.

Keywords: FLANN, SURF, mobile robot, image recognition, nearest neighbor search, Kinect, depth image, deep data pre-processing, Kinect SDK.

For citation:

Nguyen Tuan Dung, Shcherbatov I. A. Joint Recognition of the Moving and Stationary Objects in the Machine Vision Systems of Robots, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2015, vol. 16, no. 7, pp. 464—470.

DOI: 10.17587/mau.16.464-470

References

1. Navneet Dalai, Bill Triggs. Object Detection using Histograms of Oriented Gradients, available at: <http://www.webcitation.org/6DvoEuAvL> (date of access: 14.10.2014).

2. Ramisa A., Vasudevan S., Aldavert D., Toledo R., Lopez de Mantaras R. Evaluation of the SIFT Object Recognition Method in Mobile Robots, available at: http://www.iiia.csic.es/~mantaras/sift_eval.pdf (date of access: 14.10.2014).

3. Bay H., Ess A., Tuytelaars T., Van Gool L. Speeded-Up Robust Features (SURF), *Computer Vision and Image Understanding*, 2008, vol. 110, iss. 3, pp. 346—359.

4. Marron R., Janet J. C# Implementation of SLAM Using the Microsoft Kinect, available at: http://old.ee.duke.edu/files/ece/GWDD2012_Marron.pdf (date of access: 14.10.2014).

5. Shotton J., Fitzgibbon A. Cook M., Sharp T., Finocchio M., Moore R., Kipman A., Blake A. Real-time human pose recognition

in parts from single depth images, *VISAPP International Conference on Computer Vision Theory and Applications*, February 2013, pp. 119–135.

6. **Kefer M., Kubinger W.** Evaluation of Kinect Depth Sensor for Use in Mobile Robotics, *Annals of DAAAM for 2011 & Proceedings of the 22nd International DAAAM Symposium*, 2011, vol. 22, no. 1.

7. **Knuth D.** Section 6.1: Sequential Searching. Sorting and Searching. *The Art of Computer Programming 3*, Addison-Wesley, 1997, pp. 396–408.

8. **Muja M., Lowe D. G.** Fast approximate nearest neighbors with automatic algorithm configuration, *VISAPP International Conference on Computer Vision Theory and Applications*, 2009, pp. 331–340.

9. **Mount D. M., Arya S.** ANN: A Library for Approximate Nearest Neighbor Searching, available at: <http://www.cs.umd.edu/~mount/ANN/> (date of access: 05.11.2014).

10. **Connor M.** The Simple, Thread-safe Approximate Nearest Neighbor (STANN), available at: <https://sites.google.com/a/compegeom.com/stann/Home> (date of access: 05.11.2014).

11. **Forsyth D.A., Ponce J.** *Computer Vision: A Modern Approach, Robotics and Autonomous Systems*, 2006, vol. 11, no. 3, pp. 84–92.

12. **Review** information on the nearest neighbor algorithm, available at: <http://www.basegroup.ru/library/analysis/regression/knn/> (date of access 05/11/2014).

УДК 62-522.7

DOI: 10.17587/mau.16.470-475

П. А. Лошицкий, канд. техн. наук, ст. науч. сотр., p.loshitskiv@rtc.ru,

Е. Е. Шеховцова, конструктор, shehovcova@rtc.ru,

Центральный научно-исследовательский и опытно-конструкторский институт
робототехники и технической кибернетики

Расчет и моделирование работы промышленного манипулятора на силовых оболочковых элементах

На примере одного из вариантов промышленного манипулятора рассмотрена возможность успешного применения силовых оболочковых элементов типа "пневматический мускул" в качестве исполнительного привода в степенях подвижности манипулятора, а также показаны преимущества данной конструкции по сравнению с уже существующими аналогами.

Ключевые слова: силовой оболочковый элемент, пневматический привод, манипулятор

Введение

На сегодняшний день существует множество различных видов промышленных роботов, в которых применены различные типы приводов и реализуются различные кинематические схемы. В то же время разработчики современных робототехнических систем не очень активно применяют в своих разработках новые элементы приводной техники, предпочитая традиционные виды приводов. Такая ситуация существенно замедляет появление на рынке новых разработок в области автоматизации производственных процессов, обладающих лучшими

технично-экономическими показателями по сравнению с существующими аналогами.

По мнению авторов, одним из таких типов приводов, незаслуженно обделенных вниманием со стороны разработчиков, являются силовые оболочковые элементы (СОЭ) типа "пневматический мускул" (ПМ). СОЭ типа ПМ представляет собой гибкую трубчатую оболочку, армированную нерастяжимыми нитями, на концах которой закреплены присоединительные фланцы. При подаче сжатого воздуха в полость оболочки происходит ее расширение в диаметральном направлении и, как следствие, сокращение оболочки в осевом направлении. Более подробно о СОЭ типа ПМ (далее по тексту СОЭ) можно прочитать в работах [1–4].

На основе анализа силовых характеристик СОЭ (рис. 1) [1] можно выделить основные достоинства и недостатки силовых элементов.

К достоинствам можно отнести высокие значения развиваемого усилия, особенно в начальном положении, высокую удельную мощность и отсутствие сухого трения и люфтов между перемещающимися частями, что позволяет работать на ползучих скоростях. К основным недостаткам можно отнести малое значение перемещения относительно длины СОЭ и существенное падение развиваемого усилия при сокращении СОЭ. Тем не менее, несмотря на существенные ограничения, которыми обладает данный тип СОЭ, они могут успешно применяться в различной приводной технике, в том

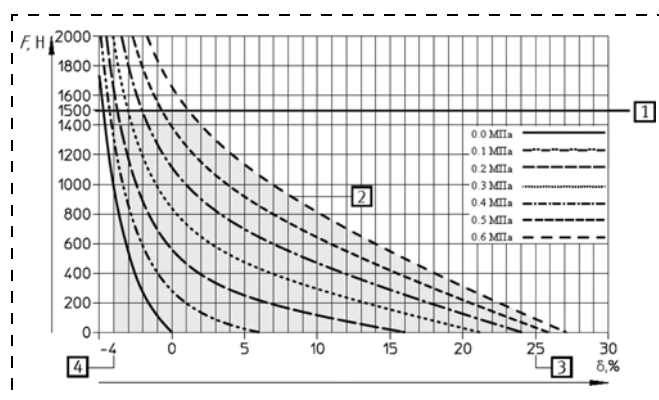


Рис. 1. Силовые статические характеристики СОЭ типа ПМ с диаметром оболочки 20 мм (F — развиваемая сила, δ — относительное сокращение СОЭ)