ИПМ им.М.В.Келдыша РАН • Электронная библиотека

Препринты ИПМ • Препринт № 24 за 2011 г.

Биндель Д., Иванов Д.С., Нуждин Д.О., Овчинников М.Ю., Трофимов С.П.

Система определения положения и ориентации макета спутника на основе блока инерциальных датчиков и звездного датчика

Рекомендуемая форма библиографической ссылки: Система определения положения и ориентации макета спутника на основе блока инерциальных датчиков и звездного датчика / Д.Биндель [и др.] // Препринты ИПМ им. М.В.Келдыша. 2011. № 24. 30 с. URL: http://library.keldysh.ru/preprint.asp?id=2011-24

ОРДЕНА ЛЕНИНА
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ им.М.В.КЕЛДЫША
РОССИЙСКОЙ АКАДЕМИИ НАУК

Д.Биндель, Д.С. Иванов, Д.О. Нуждин, М.Ю. Овчинников, С.П. Трофимов

СИСТЕМА ОПРЕДЕЛЕНИЯ ПОЛОЖЕНИЯ И ОРИЕНТАЦИИ МАКЕТА СПУТНИКА НА ОСНОВЕ БЛОКА ИНЕРЦИАЛЬНЫХ ДАТЧИКОВ И ЗВЁЗДНОГО ДАТЧИКА

Москва
2011

Система определения положения и ориентации макета спутника на основе блока инерциальных датчиков и звездного датчика. Д. Биндель, Д.С.Иванов, Д.О. Нуждин, М.Ю. Овчинников, С.П. Трофимов. Препринт ИПМ им.М.В.Келдыша РАН, Москва, 30 страниц, 24 рисунка, библиография 7 наименований.

Построены и исследованы алгоритмы определения вектора состояния тела с использованием двух независимых средств определения его ориентации и положения на горизонтальном лабораторном столе – звездная камера и блок инерциальных датчиков (двухосный акселерометр и датчик угловой скорости). Алгоритмы реализованы на макете спутника, приведены и проанализированы результаты экспериментов.

Ключевые слова: алгоритм определения ориентации, акселерометр, датчик угловой скорости, звёздный датчик

Mock-Up Position and Orientation Determination System Based on Block of Inertial Sensors and Star Tracker, D. Bindel, D.S. Ivanov, D.O. Nuzhdin, M.Yu.Ovchinnikov, S.P. Trofimov. Preprint of KIAM RAS, Moscow, 30 Pages, 24 Figures, 7 References.

Attitude and position determination algorithms based on IMU block (accelerometers and gyro sensor) and star tracker measurement are developed and investigated. Algorithms are realized on laboratory mock-up, experimental results are analyzed.

Key words: attitude determination algorithm, accelerometer, angular velocity sensor, star tracker

1. Введение

В Центре космических технологий и микрогравитации (ZARM) в г. Бремене создан стенд для отработки алгоритмов управления движением как одиночных, так и группировки спутников. Стенд состоит из гладкого стеклянного стола, двух макетов, движущихся по поверхности стола без трения благодаря воздушной подушке, имитатора Солнца и имитатора звездного неба – набора светодиодов, установленных на потолке над стеклянным столом (рис.1) [1, 2].

Рис. 1. Внешний вид стенда

На стенде реализована серия алгоритмов управления. В частности, алгоритм управления дочерним спутником в группе, обеспечивающим сохранение постоянного расстояния до главного спутника и постоянной ориентации дочернего спутника на главный спутник [3]. Был также реализован алгоритм перемещения одного аппарата в заданное положение, и алгоритм сохранения текущего положения и ориентации одного макета при воздействии возмущений [3].

Для управления макетами необходимо знать текущий вектор состояния объекта (его положение на столе, текущие скорость центра масс, ориентацию и угловую скорость). Поэтому наряду с отработкой алгоритмов управления разрабатывались алгоритмы идентификации движения макетов. Эти алгоритмы основываются на различных средствах идентификации – датчиках определения ориентации. Например, в [4] использовалась веб-камера, с помощью которой определялась ориентация и положение макета: камера устанавливалась над столом макета, на крышку макета помещали три разноцветных круга(рис.2). Снимки веб-камеры обрабатывались в сре-

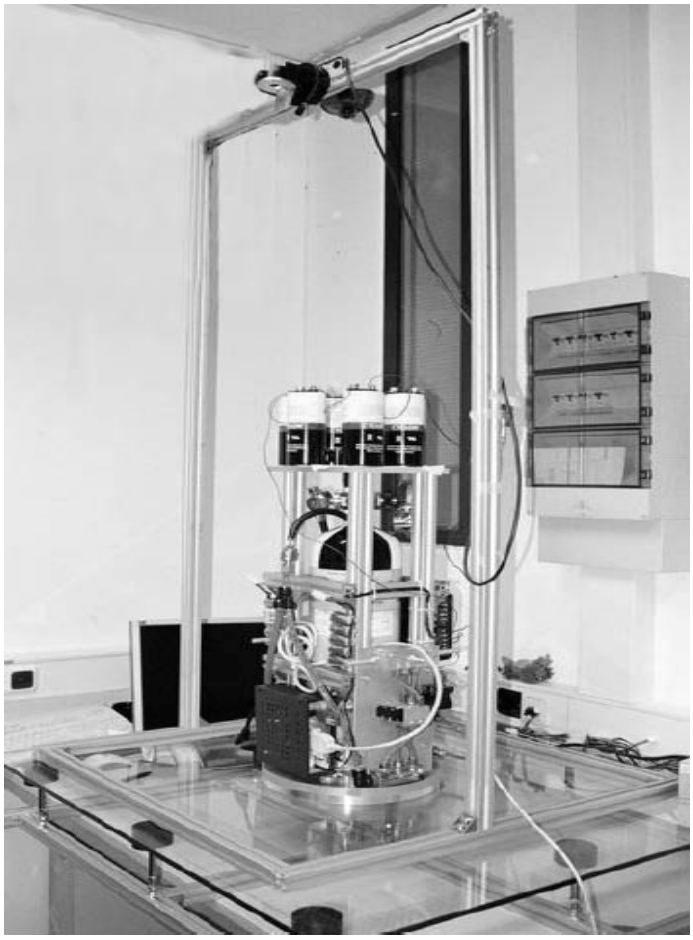

Рис. 2. Идентификация с помощью веб-камеры

де MatLab, и по положению цветных кругов на снимке определялась ориентация макета и его положение, на основе этих данных вырабатывалось управление макетом. Подобные алгоритмы используются в задачах автоматической стыковки космических аппаратов.

Для определения взаимного положения двух макетов на столе в задаче управления групповым движением спутников используются снимки главного аппарата, освещенного солнечным имитатором. На снимках распознаётся освещенная часть центрального тела, размеры которого известны, и вычисляется взаимное расстояние между объектами, а также положение и взаимная ориентация.

В настоящей работе рассматривается алгоритм определения ориентации и положения макета на столе, основанный на распознавании звезд (светодиодов) и использующий измерения блока инерциальных датчиков – датчика угловой скорости и двух инклинометров. Теоретически каждое из средств идентификации движения способно отслеживать текущее состояние объекта, однако особенности каждого из средств обуславливают собой недостатки при использовании только одного из них (звездного датчика или блока инерциальных датчиков). А использование сразу двух этих средств позволяет избавиться от этих недостатков и улучшить точность определения движения макета на столе.

В настоящей работе сначала рассматривается подробно каждое из средств идентификации, исследуются их точностные характеристики, далее приводится алгоритм, использующий оба средства, анализируются результаты отработки этого алгоритма на стенде.

2. Алгоритм идентификации участка звездного неба

2.1. Классификация алгоритмов

После изобретения первой звездной камеры (*Salomon*, 1976) на основе передовой тогда технологии ПЗС (приборов с зарядовой связью) было разработано множество алгоритмов идентификации участков звездного неба. Такие алгоритмы находят применение главным образом в задаче определения ориентации космического аппарата. Все их можно классифицировать по нескольким параметрам:

1. использование априорной информации
   a) рекурсивные алгоритмы – делается предположение о том, какие звезды должны попасть в поле зрения камеры;
   b) алгоритмы *“lost-in-space”* – не используется никакая априорная информация;
2. использование информации о яркости звезд;
3. метод выбора звезд для извлечения информации об участке неба на снимке
   a) звезды выбираются произвольно;
   b) выбор осуществляется по определенному критерию (яркость, угловое расстояние и т.п.);

4. калибровочная инвариантность – инвариантность по отношению к дисторсиям первого порядка.

Стандартный цикл процесса идентификации участка звездного неба показан на рис.3.

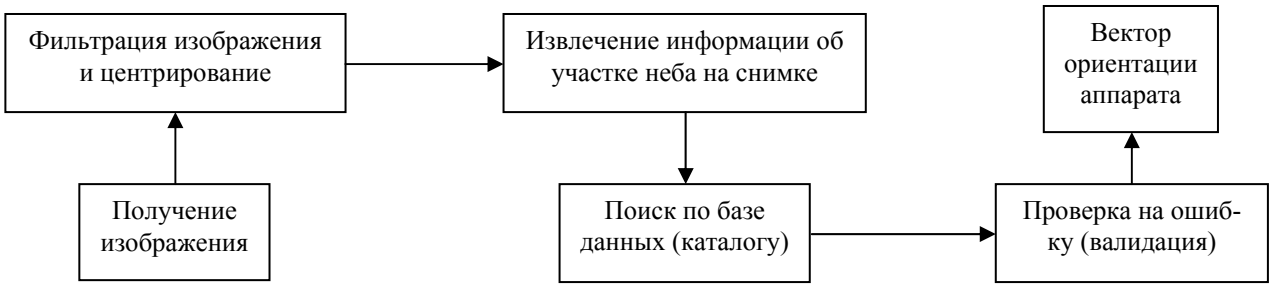

Рис.3. Стандартный цикл процесса идентификации участка звездного неба

2.2. Особенность разработанного алгоритма

На выбор того или иного метода идентификации звезд для моделирования на стенде в ZARM были наложены определенные ограничения:

1) невозможность табулирования межзвездных угловых расстояний, меняющихся с изменением удаления камеры от плоскости “неба”;

2) отсутствие различий в яркости между “звездами”;

3) использование в качестве звездной камеры обычной веб-камеры, подверженной различным дисторсиям и требующей периодической рекалибровки.

Кроме того, недостаточная предсказуемость “орбитального” движения (движения макета по столу) делала трудноосуществимым применение рекурсивных алгоритмов. Как следствие всех факторов, был выбран калибровочно-инвариантный *“lost-in-space”* алгоритм, основанный на триангуляции Делоне.

Триангуляцией Делоне (далее – ТД) для множества точек на плоскости называют такую триангуляцию, что никакая точка не содержится внутри окружности, описанной вокруг любого треугольника с вершинами в точках данного множества (рис.4). Основным свойством ТД является максимизация минимального угла среди всех углов всех построенных треугольников. Таким образом, избегаются “тонкие” треугольники. Помимо этого, для заданного набора точек ТД – единственная.

Идея применения ТД в задаче идентификации звезд не новая [5]. Еще раньше была предложен первый калибровочно-инвариантный алгоритм, подразумевающий использование внутренних углов в образованных звездами треугольниках взамен межзвездных угловых расстояний [6]. Для моделирования на стенде в ZARM с учетом всех особенностей и ограничений было решено использовать синтез этих двух алгоритмов.

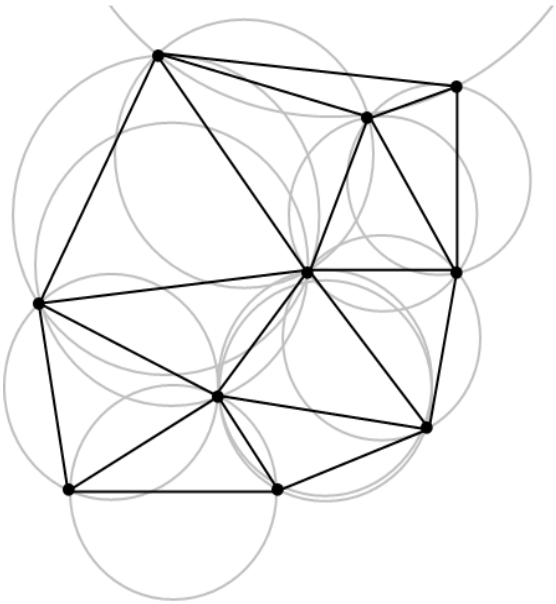

Рис.4. Пример выполненной триангуляции Делоне

Именно единственность ТД позволяет использовать ее как инструмент в алгоритме идентификации участка звездного неба: для всех звезд неба с яркостью выше пороговой для данной камеры (для стенда в ZARM – для всех “звезд” на “небе”) строится единственно возможная ТД, то есть на основе звездного каталога создается база данных, содержащая информацию о всех треугольниках из ТД. Под информацией для каждого треугольника можно понимать, например, номера (координаты) звезд, образующих треугольник, и два внутренних угла - наибольший и наименьший - в треугольнике. Далее, при условии, что в поле зрения камеры попадает достаточно большое количество звезд, б*о*льшая часть треугольников из ТД полученного снимка будет присутствовать в базе данных о ТД для всего неба (за исключением, быть может, треугольников, содержащих звезды вблизи границ кадра). Такие треугольники в дальнейшем будем называть правильными. Руководствуясь взятой из базы данных информацией о каждом распознанном на снимке треугольнике, можно получить массив предполагаемых координат для некоторой фиксированной точки снимка (например, центра кадра). Если распознанный треугольник – правильный, то соответствующий элемент массива предполагаемых координат выбранной точки будет близок к ее истинным координатам. Таким образом, выбирая из массива подмассив максимальной длины, элементы которого близки друг к другу, можно с достоверностью утверждать, что среднее значение элементов такого подмассива будет лишь незначительно отличаться от истинных координат выбранной точки. Подобные рассуждения и составляют основу выбранного метода идентификации.

Будем использовать стандартный символ $O(...)$ для худшей асимптотической оценки члена максимального порядка в выражении для вычислительных затрат на тот или иной этап цикла, а также для оценки объема памяти, требующейся для хранения базы данных. Буква n будет обозначать число звезд в звездном каталоге, а f – среднее число звезд, попадающих в поле зрения камеры. Кроме того, везде далее $\lg n$ означает $\log_2 n$.

2.3. Описание алгоритма

Рассмотрим теперь подробно каждый этап цикла идентификации с указанием его вычислительной сложности.

2.3.1. Нахождение координат центров звёзд

Начальные этапы получения изображения и его фильтрации и центрирования (нахождения пиксельных координат “центров” звезд) стандартны для всех типов алгоритмов. Кратко опишем их.

Получение снимка происходит путём оцифровывания выходных значений элементов ПЗС-матрицы, в результате чего имеется массив информации об интенсивности света в определённом диапазоне длин волн (в нашем случае в видимом диапазоне) в каждом пикселе чувствительной матрицы. Далее определяется уровень шума снимка, производится фильтрация: находится максимальная и минимальная интенсивности, они нормируются, после чего устанавливается значение, ниже которого значение интенсивности пикселя приравнивается нулю. Затем производится поиск центра звёзд [7]: находятся расположенные рядом друг с другом пиксели с высокой интенсивностью и рассчитываются координаты центра светимости звезды $\boldsymbol{x}_c$, $\boldsymbol{y}_c$ на снимке по формуле

$$x_c = \frac{\sum\limits_{i=1}^{N} x_i I_i}{\sum\limits_{i=1}^{N} I_i}, \quad y_c = \frac{\sum\limits_{i=1}^{N} y_i I_i}{\sum\limits_{i=1}^{N} I_i},$$

где $\boldsymbol{N}$ – общее количество пикселей в рассматриваемой области x_i – x-координата *i*-го пикселя, y_i – y-координата *i*-го, I_i – интенсивность *i*-го пикселя.

Далее следует этап извлечения информации об участке неба. Но сначала нужно сказать о структуре базы данных.

2.3.2. Структура базы данных

Помимо звездного каталога, стандартная запись которого представляет номер звезды и ее координаты (в случае плоской имитации звездного неба – декартовы координаты “звезды” на плоскости), для рассматриваемого алгоритма требуется также база данных, содержащая информацию о треугольниках, входящих в ТД всего неба. Запись в такой базе данных состоит из номеров звезд, образующий треугольник, и величин наибольшего и наименьшего внутренних углов треугольника. Размер такой базы данных определяется числом треугольников в ТД неба и поэтому имеет тот же порядок $O(n)$, что и сам звездный каталог.

2.3.3. Извлечение информации об участке неба на снимке

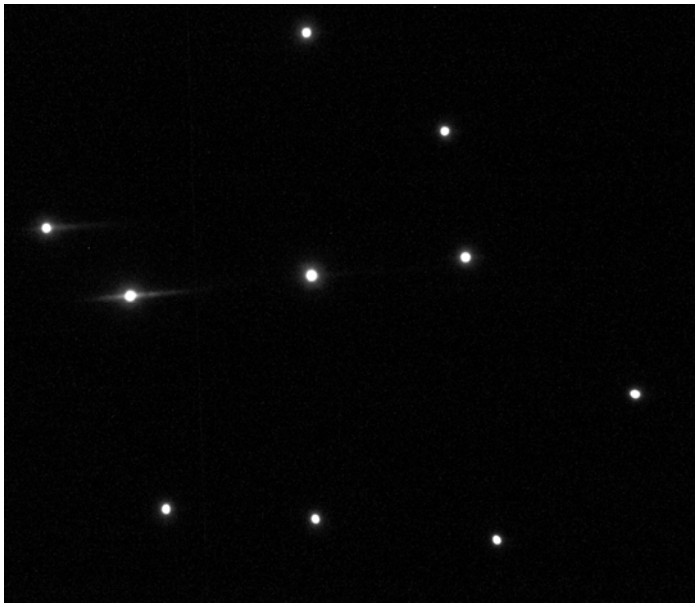

Рис. 5. Пример участка звёздного неба

Для извлечения информации об участке неба, попавшем в поле зрения камеры (рис. 5), выполняется ТД этого участка. Далее для каждого полученного треугольника вычисляются значения наибольшего и наименьшего внутренних углов. Наименьшая вычислительная сложность, достигающаяся, к примеру, в алгоритмах построения ТД методом слияния, равна $O(f \lg f)$. Как уже упоминалось, при достаточно большом f большинство треугольников из триангуляции участка неба (за исключением, быть может, треугольников, содержащих звезды вблизи границ кадра) будут правильными. Это следует из основного свойства ТД: избегаются “тонкие” треугольники, в которых одна из вершин сильно удалена от других.

2.3.4. Поиск по базе данных (каталогу)

Производится поиск каждого треугольника из ТД участка неба среди всех треугольников из базы данных по найденным величинам наибольшего и наименьшего внутренних углов. Треугольник считается идентифицированным, если отклонение по обоим углам от значений для некоторого треугольника в базе данных не превосходит определенный предел. Тогда звезды идентифицированного треугольника отождествляются со звездами, номера которых указаны в записи для соответствующего треугольника в базе данных. Матрица аффинного преобразования, переводящего пиксельные координаты звезд на снимке в реальные координаты отождествленных с ними звезд из звездного каталога, отыскивается из решения переопределенной системы линейных уравнений методом наименьших квадратов. Исходя из этого, находятся предполагаемые координаты некоторой фиксированной точки снимка (в выполненном на стенде эксперименте – центра кадра) и записываются в специальный массив.

Следует еще раз подчеркнуть, что поиск производится среди *всех* треугольников из базы данных. Таким образом, треугольник из ТД участка неба может быть идентифицирован много раз.

Вычислительная сложность поиска определяется произведением чисел треугольников в ТД участка неба и ТД всего неба и, следовательно, равна $O(fn)$.

2.3.5. Проверка на ошибку (валидация)

Последним по порядку (но не по важности) этапом является так называемый этап валидации, сводящийся для рассматриваемого алгоритма к нахождению истинных координат выбранной фиксированной точки снимка на основе полученного массива ее предполагаемых координат. Как указывалось выше, для получения достоверного результата достаточно выбрать максимальной длины подмассив близких значений, так как именно правильные (и правильно идентифицированные!) треугольники дадут скореллированные предполагаемые значения координат выбранной точки. Поэтому ясно, что алгоритм устойчив по отношению к ошибкам идентификации треугольников из ТД участка неба, главное – это наличие некоторого количества правильно идентифицированных треугольников. Вычислительная сложность этапа валидации определяется в основном процессом сортировки массива предполагаемых координат размером $O(f)$ и в худшем случае имеет порядок $O\left(f^2\right)$.Определение координат центра кадра есть фактически нахождение ориентации оптической оси камеры в пространстве. В случае необходимости можно также добавить в алгоритм определение угла поворота камеры вокруг этой оси.

Основные этапы алгоритма собраны воедино в блок-схему (рис.6). Под началом его работы надо понимать момент, когда известны пиксельные координаты звезд на снимке. Через N обозначено количество треугольников в ТД участка неба, а через M – количество треугольников в ТД всего неба (т.е. количество треугольников в базе данных). На потолке в лаборатории над столом было установлено 180 светодиодов. Каталог треугольников Делоне после триангуляции для всего неба состоял из 322.

2.4. Реализация алгоритма

Алгоритм был реализован в среде MatLab. В качестве звездной камеры использовалась веб-камера производства фирмы Philips с размером ПЗС-матрицы 640x480 пикселей. Камера устанавливалась на верхней части макета как показано на рис. 7. Так как бортовой компьютер макета лимитирован по объёму вычислений (установлен относительно слабый процессор), то данные, поступающие с веб-камеры, обрабатывались на ноутбуке, установленном также на верхней части макета. После отработки алгоритма, на бортовой компьютер посылалась информация о местоположении макета в системе координат, связанной со столом, и угле поворота системы координат, связанной с макетом, относительно этой системы координат. После получения

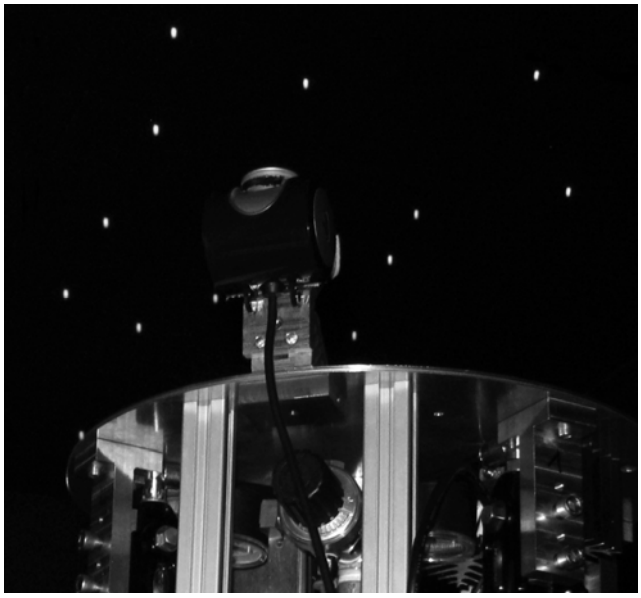

Рис. 7. Установка веб-камеры

двух измерений линейные скорости и угловая скорость вращения макета вычислялись дифференцированием методом конечных разностей. Таким образом, алгоритм выдавал полный шестимерный вектор состояния.

Исследуем характеристики реализованного алгоритма определения положения и ориентации макета.

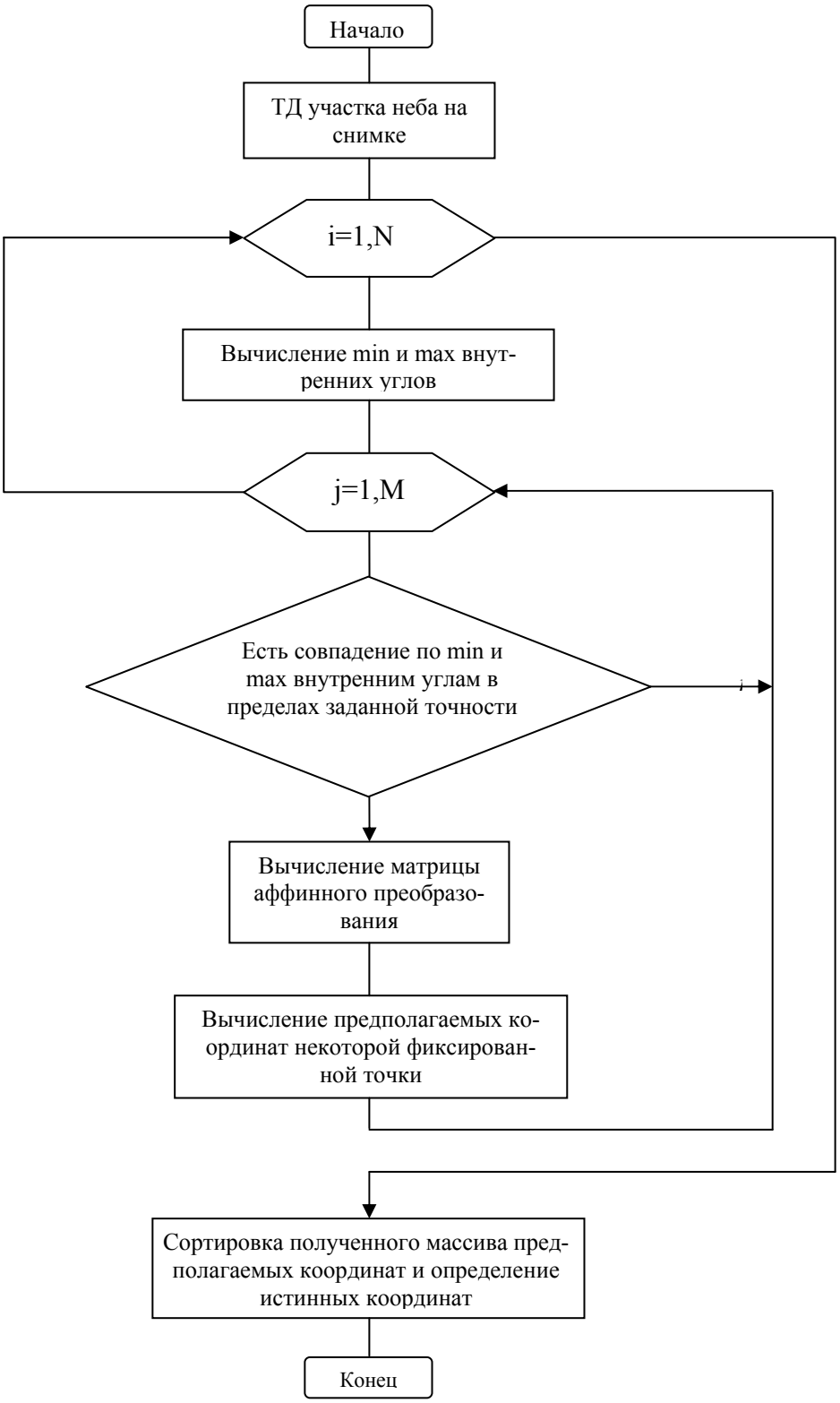

Рис.6. Блок-схема алгоритма

2.4.1. Быстродействие алгоритма

Под быстродействием алгоритма будем понимать количество времени, необходимое для определения положения и ориентации макета. Для исследования этой характеристики достаточно провести серию экспериментов по определению положения макета при движущемся макете. На рис. 8 изображены результаты одного из таких экспериментов, в котором макет двигался примерно по периметру стола с постоянной по модулю скоростью. На рис. 8 также изображены серыми стрелками линейные скорости макета во время движения. На рис. 9 изображен график зависимости времени, потраченного алгоритмом на определение текущего положения, от времени в эксперименте.

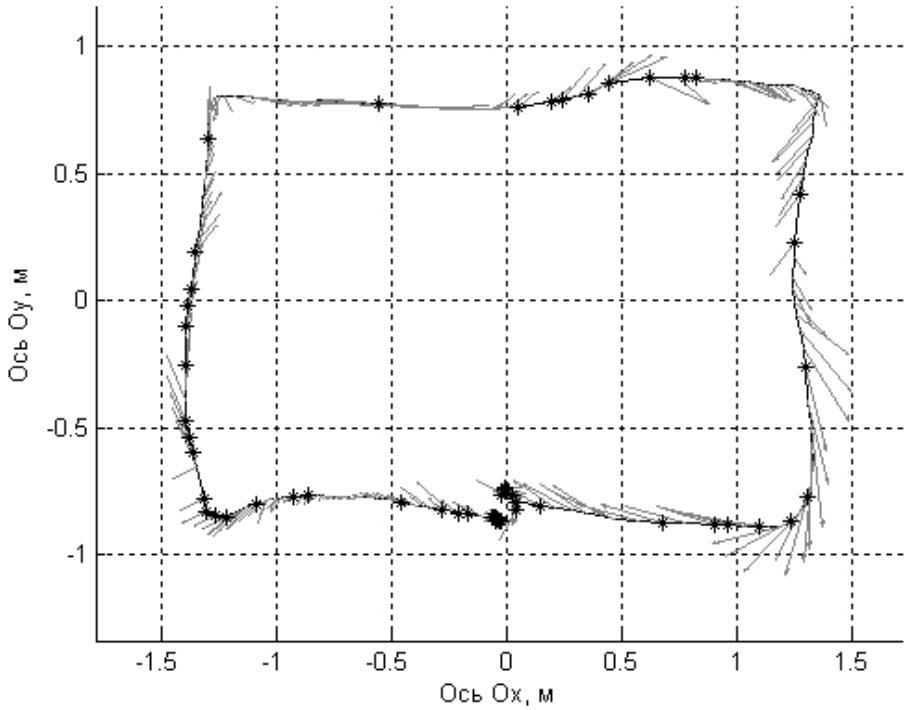

Рис. 8. Определение положения макета во время движения примерно по периметру стола. Серыми стрелками обозначены направления вектора скорости, Звёздочками обозначены положения макета, в которых возникли проблемы с определением ориентации

Как видно из рис.9, в среднем, если алгоритм сумел определить ориентацию по снимку, сделанному веб-камерой, то на полный цикл определения, включающий в себя получение снимка, его фильтрацию и обработку, уходит примерно 0.6 секунд. Это время незначительно отличается от снимка к снимку, и это отличие зависит от количества звезд, попавших в поле зрения кадра, от положения треугольников ТД в базе данных (тратится различное время на поиск), и так далее (см. раздел 2.3). Однако, как видно из рис.9, иногда возникает ситуация, что алгоритм не может определить текущее положение макета. Это возникает по причине того, что все предполагаемые координаты макета, вычисленные на основе каждого треугольника ТД участка звездного неба, отличаются на величину, превышающую заданную ошибку. Другими словами, не оказалось хотя бы двух близких значений координат. Перечислим основные причины, по которым возникает такая ситуация.

- Участки звездного неба со слишком малым количеством звезд. Когда макет находится под таким участком, может случиться так, что ТД участка звездного неба даст на выходе только один треугольник, входящий в базу данных треугольников, тогда как для успешного определения положения необходимо иметь хотя бы два треугольника на участке звездного неба.
- Высокая скорость движения макета по столу. При большой скорости движения снимки звезд смазываются и, как следствие, центр светимости звезды определяется с ошибкой. Следовательно, треугольники ТД вычисляются с некоторой ошибкой, возможна даже ошибочная триангуляция. Всё это приводит к тому, что в базе данных вместо нужного треугольника находится другой, более подходящий по значениям ошибочно рассчитанных углов.

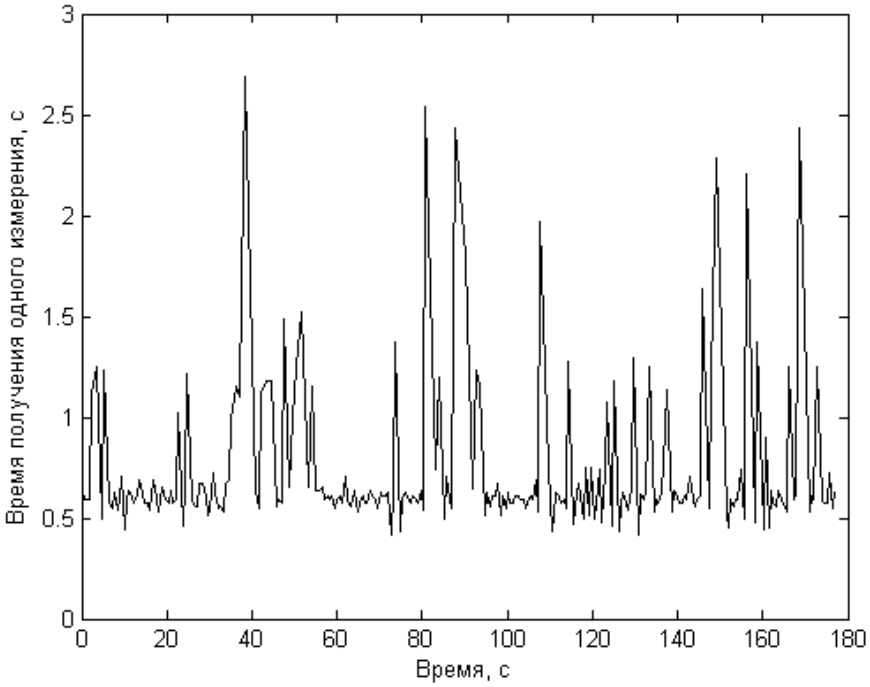

Рис. 9. Зависимость времени, потраченного на определение положения, от времени эксперимента

На рис.9 отчетливо видно «квантование» времени, ушедшего на определение положения макета в данный момент времени. Максимально на определение положения ушло 2.7 секунд, что примерно равно по времени обработке 4 снимков звездного неба (каждый по 0.6 секунд). Несмотря на достаточно большое количество точек, в которых алгоритм не смог определить положения, среднее время определения составляет 0.72 секунды.

На рис.10 звездочками примерно обозначены те положения макета, в которых возникли проблемы с определением положения. Достаточно сложно сказать, по какой из двух вышеперечисленных причин в каждой из точек происходил сбой алгоритма, однако с большой долей вероятности можно утверждать, что над точкой (0, -0.8), когда макет стоял неподвижно, слишком мало звезд для определения положения макета.

2.4.2. Точность алгоритма

Оценим точность работы алгоритма определения положения и ориентации макета. Это можно сделать несколькими способами. Например, можно проверить работу алгоритма в состоянии покоя. На рис.10 изображены данные об одной из координат макета, получаемые в состоянии его покоя.

Наилучшая точность определения положения составляет порядка одного миллиметра. Сложнее определить точность работы алгоритма в состоянии движения, так как во время движения макета изображения “звезд” смазываются, что влияет, прежде всего, на точность вычисления их центров светимости, а, значит, и на определение положения. Легко понять, что чем быстрее движется макет во время получения снимка, тем хуже точность. Более того, при скорости, выше определённой, ошибки определения центров звёзд будут таковыми, что алгоритм не сможет найти среди базы данных полученные треугольники Делоне, то есть алгоритм будет неработоспособен.

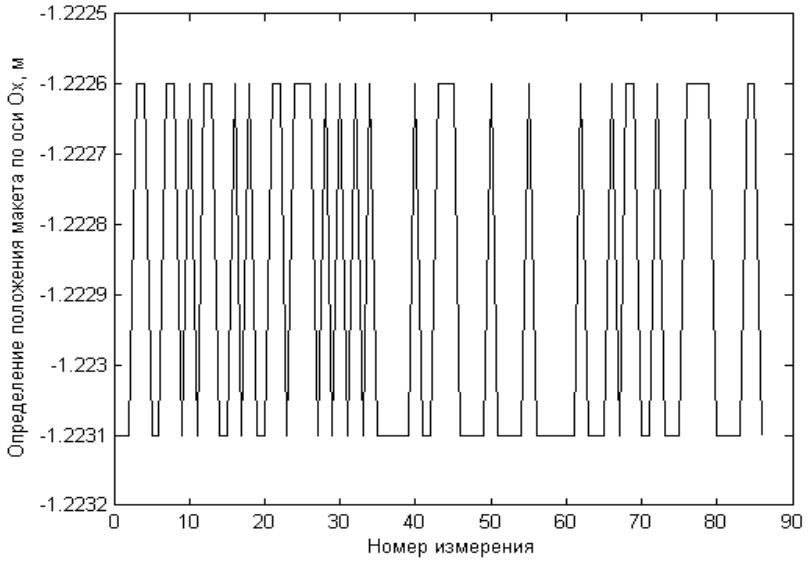

Рис.10. Точность определения положения в состоянии с нулевой скоростью

3. Алгоритм определения положения и ориентации с помощью инерциальных датчиков

Зная начальные условия (вектор состояния в начальный момент времени) и получая измерения угловой скорости и ускорений по двум осям, можно вычислять путём их интегрирования вектор состояния макета в каждый момент времени, когда приходят измерения с датчиков. Построим алгоритм, основанный на измерениях инерциальных датчиков. Но сначала рассмотрим проблемы, возникающие при их использовании.

3.1. Недостатки использования инерциальных датчиков

При использовании датчика угловой скорости и акселерометров для вычисления текущего вектора состояния тела могут возникать ошибки. Разберем природу этих ошибок и способы их устранения.

- *Шум датчиков*. Вследствие шума измерений, возникающего при оцифровывании аналогового сигнала с датчиков (а также вызванного дробовым и тепловым шумами), возникает неточность вектора состояния, вычисленного на основе зашумлённых данных. От шума полностью избавиться нельзя, но можно уменьшить его величину, например, путём усреднения нескольких измерений или с помощью фильтрации.
- *Неточность знания начальных данных*. Для интегрирования важно знать начальное положение (вектор состояния), и в случае неточности задания начальных условий при интегрировании возникает постоянная ошибка, равная ошибке задания начальных условий. К сожалению, от этой ошибки тоже полностью избавиться нельзя (так как начальные условия измеряются также каким-либо датчиком, у которого есть шум), однако, если периодически в процессе движения получать неточные начальные данные и если предположить, что математическое ожидание шума начальных данных равно нулю, то можно постепенно уменьшать постоянную ошибку при интегрировании.
- *Ошибка знания нуля измерений датчика*. Нулем измерений (или смещением ноля) называется выходное значение датчика, когда он покоится относительно некоторой инерциальной системы координат (для датчика угловой скорости), либо когда чувствительная ось совпадает с направлением, относительно которого ведётся измерение (для инклинометров это направление местной вертикали). При неточности знания этого параметра, при интегрировании будем получать смещенное значение скорости и увеличивающуюся ошибку в положении. Смещение ноля следует находить с помощью калибровки датчиков. Для некоторых инерциальных датчиков (ДУС) смещение ноля зависит от температуры, поэтому перед испытаниями следует найти зависимость ноля от температуры.
- *Ошибка установки датчиков.* Эти ошибки также ведут к смещенным на некоторую величину измерениям. Устранение таких ошибок также решается с помощью калибровки датчиков после их установки в рабочее положение на макете. Кроме того, инклинометры следует поместить в точке, находящейся на одной вертикали с центром масс, так как в противном случае при вращении макета на измерения датчиков будет влиять центробежное ускорение.

3.2. Алгоритм определения вектора состояния

Для вычисления вектора состояния макета в текущий момент времени по измерениям инерциальных датчиков был разработан следующий алгоритм, который можно формально разделить на два этапа.

3.2.1. Предварительная калибровка

Для того чтобы найти нули датчиков, а также ошибку их установки, перед экспериментом следует провести предварительную калибровку. Это необходимо делать каждый раз, так как значение ноля инклинометров изменяется от включения к включению.

Калибровка производится следующим образом. Перед включением программы управления проводятся измерения акселерометров в состоянии покоя макета. Так как в состоянии покоя на макет не действуют управляющие воздействия, макет находится на плоской поверхности стола, то выходные значения датчиков в этот момент являются значением ноля. Кроме того, в смещение ноля будет входить ошибка установки блока инерциальных датчиков на корпус макета. Таким образом, с помощью предварительной калибровки убираются 3 и 4 недостатки использования инклинометров, описанные выше.

3.2.2. Интегрирование измерений

Обработка данных и интегрирование производится с помощью микроконтроллера и бортового вычислителя. Сначала микроконтроллер опрашивает поочерёдно датчик угловой скорости и двухосный инклинометр и получает байты, в которых зашифрованы значения, пропорциональные угловой скорости и ускорениям по двум осям соответственно. Эти значения вычисляются из полученных байтов согласно документации, сопровождающей эти датчики. На микроконтролере также производится усреднение получаемых данных, что приводит к некоторому уменьшению шума датчиков. При передаче измерений с микроконтроллера на бортовой компьютер производится суммирование всех передаваемых данных и пересыл этого значения на БК, где по это проверяется и делается вывод о правильности передачи данных.

Рис.11. Блок схема алгоритма

После передачи данных на бортовой компьютер производится сначала их калибровка, то есть из принимаемых измерений вычитаются значения смещения ноля, полученные на этапе предварительной калибровки. После этого полученные измерения ускорений пересчитываются из собственной системы координат в абсолютную по формулам:

$$a_X = a_x \cos\varphi + a_y \sin\varphi,$$

$$a_Y = a_y \cos\varphi - a_x \sin\varphi.$$

Здесь a_X, a_Y - ускорения в инерциальной системе координат, связанной со столом, a_y - измерения в собственной системе координат, связанной с макетом, φ - угол поворота связанной системы координат в инерциальную, полученный интегрированием угловой скорости макета.

После получения значений a_X, a_Y производится проверка на аномальные измерения, которые иногда имеют место возможно из-за редких сбоев электроники датчиков. Если значения a_X, a_Y превышают 0,5 м/с^2, а измерения угловой скорости ω превышают 50 град/с, то они не рассматриваются, интегрирование не проводится, а программа ждёт приема следующих значений.

После этого производится интегрирование по следующим формулам:

$$\begin{cases} v_X^i = v_X^{i-1} + a_X^i (t_i - t_{i-1}), \\ v_Y^i = v_Y^{i-1} + a_Y^i (t_i - t_{i-1}), \end{cases}$$

$$\begin{cases} X^i = X^{i-1} + v_X^i (t_i - t_{i-1}) + \dfrac{a_X^i}{2} (t_i - t_{i-1})^2, \\ Y^i = Y^{i-1} + v_Y^i (t_i - t_{i-1}) + \dfrac{a_Y^i}{2} (t_i - t_{i-1})^2, \\ \varphi^i = \varphi^i + \omega^i (t_i - t_{i-1}). \end{cases}$$

Здесь v_X^i, v_Y^i - компоненты скоростей макета в инерциальной системе координат, X^i, Y^i, φ^i - координаты центра масс макета в инерциальной системе координат и угол поворота собственной системы координат в инерциальную соответственно в момент приёма измерений t_i, t_{i-1} - момент приема предыдущих измерений. Таким образом, на выходе в каждый момент t_i имеем вектор состояния макета, который имеет следующий вид:

$$\begin{bmatrix} X^i & v_X^i & Y^i & v_Y^i & \varphi^i & \omega^i \end{bmatrix}.$$

3.3. Реализация алгоритма

Описанный выше алгоритм определения ориентации и положения был реализован на макете. Рассмотрим аппаратные средства, с помощью которых был реализован алгоритм, и характеристики используемых датчиков.

3.3.1. Описание блока датчиков

Для удобства использования и компоновки датчиков на корпусе макета был создан блок инерциальных датчиков или, по другому, блок IMU (Internal Mea-

surement Unit). Он изображен на рис. 12. Блок IMU состоит из двух датчиков и микроконтроллера.

В качестве датчика угловой скорости используется датчик ADIS 16100 производства фирмы Analog Devices. Этот датчик имеет диапазон измерений $\pm 300^\circ / c$. Точность составляет порядка $\sigma = \pm 0.1^\circ / c$ при частоте выдачи измерений порядка $f = 1000\,Гц$. Смещение ноля линейно зависит от температуры в диапазоне от $-40^\circ C$ до $+85^\circ C$.

Двухосный инклинометр SCA103T-D04 имеет диапазон измерений $\pm 0.26\,g = \pm 2.55 м / c^2$. Точность определения ускорения составляет $\sigma = \pm 0.002 м / c^2$ при частоте выдачи измерений порядка $f = 1000\,Гц$. Типичная зависимость смещения ноля датчика от температуры составляет $\pm 0.0003 м / c^2 / ^\circ C$. Рабочий диапазон температур $-40^\circ C$ до $+125^\circ C$.

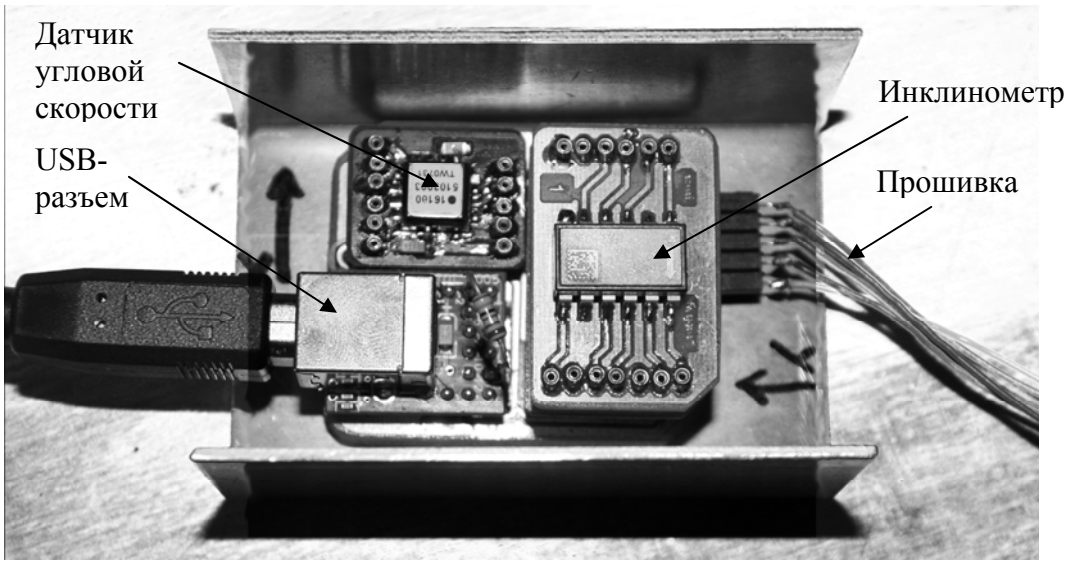

Рис. 12. Внешний вид блока IMU

Микроконтроллер PIC16F876A опрашивает инерциальные датчики с частотой $f = 1000\,Гц$, производит усреднение измерений и отправляет их на бортовой компьютер через USB порт.

Блок IMU помещен в металлический корпус, который жестко закреплён на корпусе макета, в точке, располагающейся на одной вертикали с центром масс макета (это необходимо, чтобы акселерометры не измеряли центростремительное ускорение при вращении макета). Блок обеспечивается питанием 5 Вольт, поступающим по USB-разъему.

3.3.2. Результаты экспериментов

Рассмотрим несколько экспериментов по определению вектора состояния макета. Сначала построим графики измерений ускорений и угловой скорости.

Как видно из рис. 13 измерения акселерометра по оси *ox* собственной системы координат достаточно сильно шумят, но выделить управляющие воздействия со стороны импульсных двигателей возможно. Кроме шума измерений акселерометр улавливал некоторые периодические (с периодом примерно 1с), очень кратковременные импульсы довольно сильной амплитуды $\pm 0,07 м / c^2$,

что почти в два раза превышает амплитуду управляющих импульсов по оси *ox* собственной системы координат ($\pm 0,04 м / с^2$). Возможно эти очень кратковременные импульсы никак не связаны с движением макета, а являются, например, измеренными периодическими колебаниями пола лаборатории, на котором стоит стол. Скорее всего эти всплески – это неидеальность клапанной системы импульсных двигателей, которые на несколько миллисекунд периодически открывались.

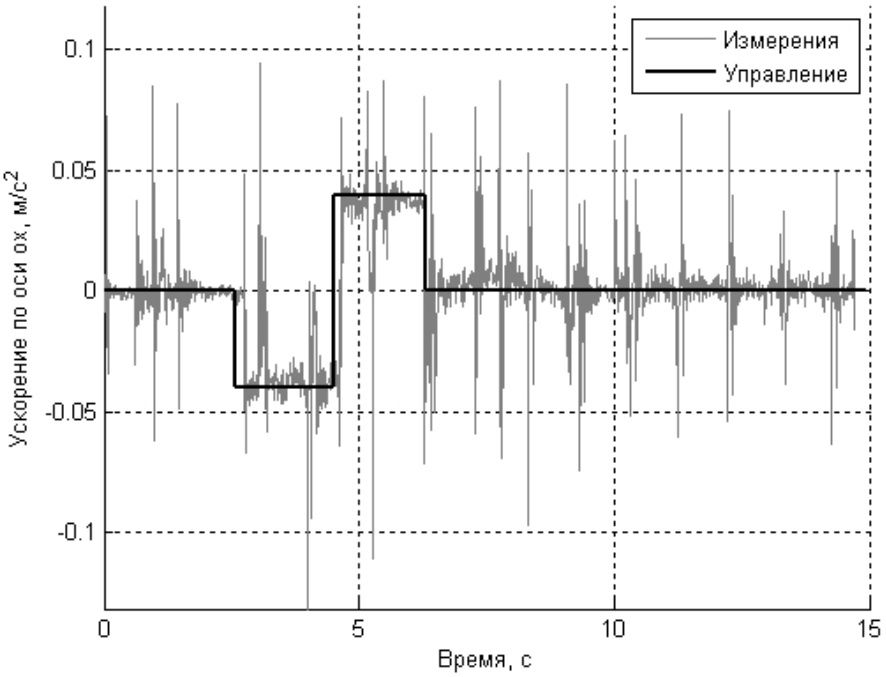

Рис. 13. Измерения ускорения

На рис. 14 изображен пример измерения угловой скорости. На этом отрезке времени было проведено два управляющих воздействия.

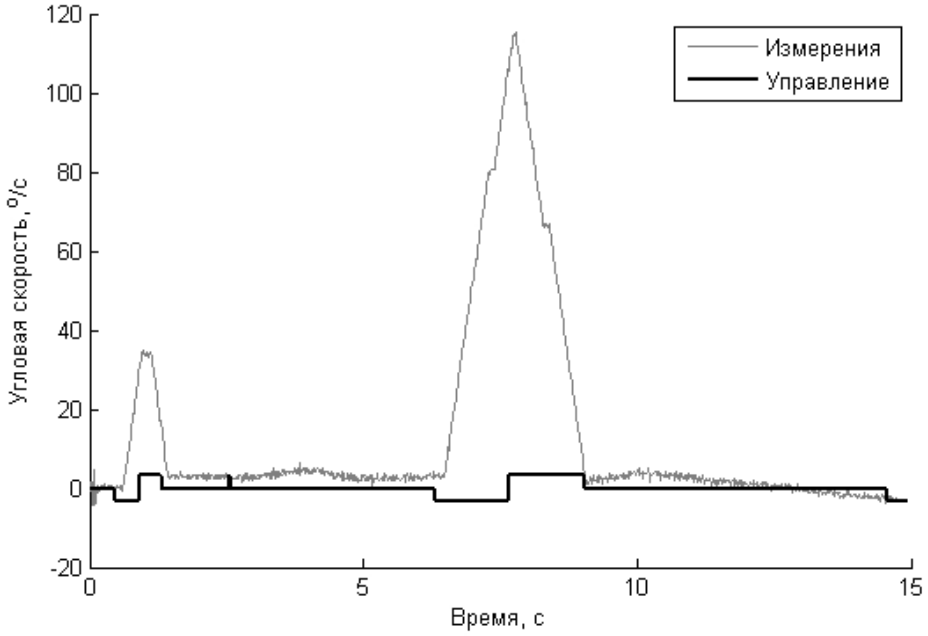

Рис. 14. Измерения угловой скорости

Для сравнения результатов работы алгоритма определения ориентации на основе интегрирования измерений сравним их с данными, полученными с помощью звездной камеры.

На рис. 15 представлены результаты интегрирования угловой скорости и для сравнения результаты, полученные на основе звездной камеры. Как видно, данные неплохо совпадают, только измерения по звездной камере запаздывают примерно на 0.6 с. Это связано с тем, что звездная камера тратит примерно это время на расчет положения и ориентации.

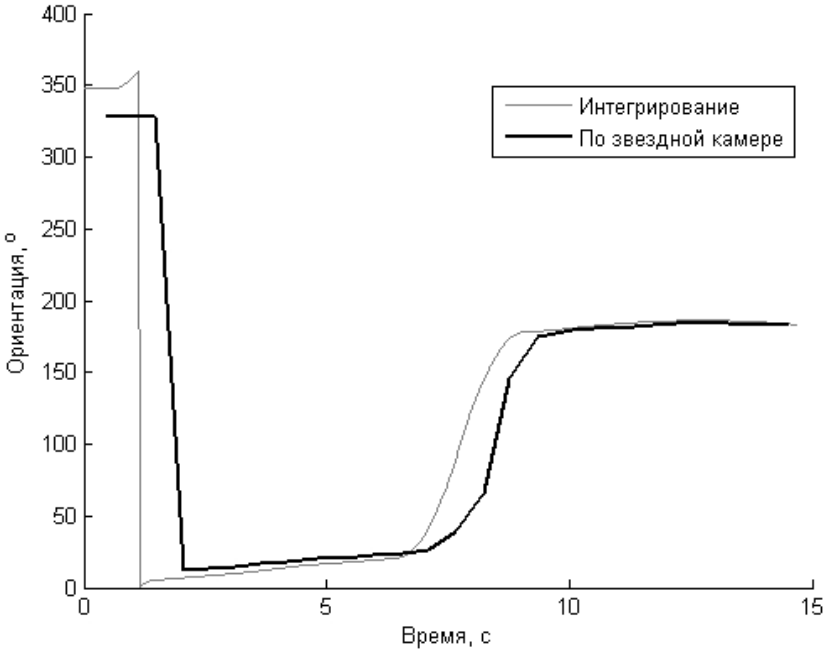

Рис. 15. Ориентация макета

На рис. 16 и 17 изображены графики изменения скоростей и положения макета в инерциальной системе координат от времени, полученные с помощью интегрирования измерений и с помощью звездной камеры.

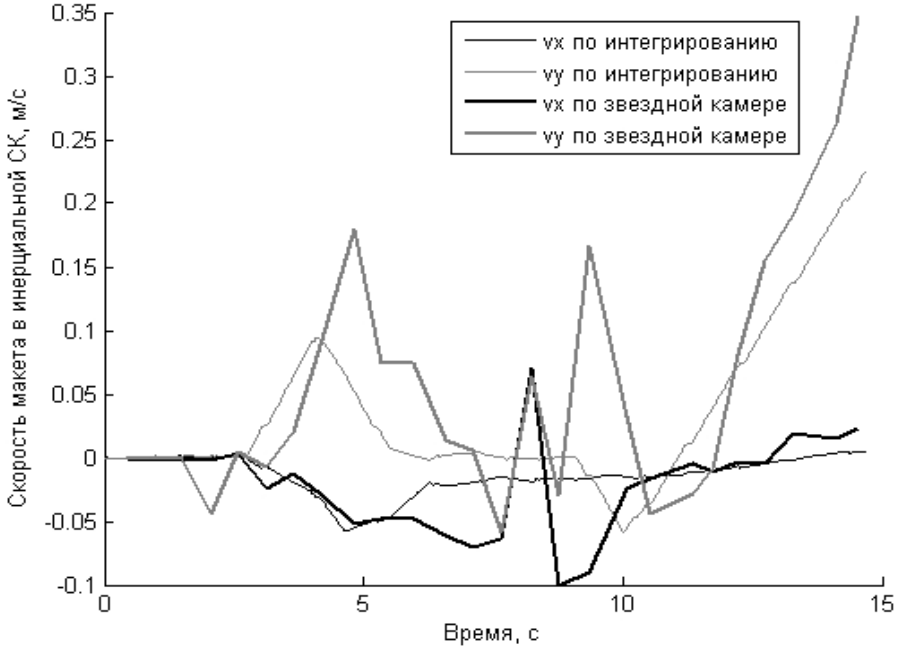

Рис. 16. Компоненты скоростей макета

Как видно из графиков компонент скоростей, данные, полученные по звездной камере, имеют сильно большую ошибку (так как вычислялись дифференцированием по двум точкам положения), а данные, полученные с помощью интегрирования − более плавные и близки к кривым, полученным по звездной камере. Однако на рис. 17 положение макета, полученное с помощью интегрирования и на основе звездной камеры, начинают расходиться. Это можно объяснить неточностью определения смещения ноля инклинометров на этапе калибровки. Ошибка в смещении ноля на $\pm 0,001 м/с^2$ приводит к конечной ошибке положения за 10 с, равной $\delta = at^2/2 = \pm 0,05 м$, а точность определения ускорения составляет $\sigma = \pm 0.002 м/с^2$. Если же во время предварительной калибровки произойдет один из кратковременных импульсов, изображенных на рис. 13, то ошибка определения смещения ноля может доходить до $\pm 0,005 м/с^2$. Эта ошибка опасна при интегрировании наращиванием линейной скорости, которой нет на самом деле, что приводит к дрейфу положения, который можно увидеть на рис. 17.

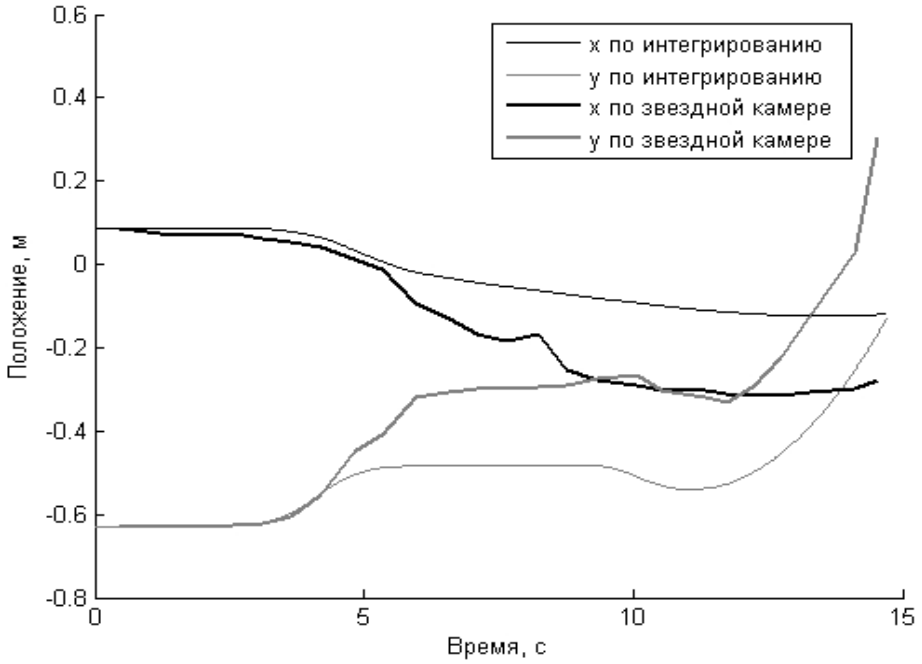

Рис. 17. Положение макета

3.3.3. Нахождение смещения ноля инклинометра

Выше мы предположили, что сильное несоответствие между проинтегрированным положением и данными по звездной камере вызваны неточным знанием ноля инклинометров. Проверим это предположение. Найдем с помощью метода наименьших квадратов такие смещения ноля по двум осям, которые будут минимизировать квадрат разницы между вычисленными значениями. Другими словами, возьмем функционал

$$\Phi = \sum_{i=1}^{N} \left[\left(x^{u}(t_i) - x^{зв}(t_i) \right)^2 + \left(y^{u}(t_i) - y^{зв}(t_i) \right)^2 \right], \qquad (1)$$

где $x^{зв}(t_i)$, $y^{зв}(t_i)$ – это координаты центра масс макета в инерциальной системе координат по данным звездной камеры, полученные в момент времени t_i, $x^u(t_i)$, $y^u(t_i)$ – это координаты центра масс макета, полученные интегрированием на момент времени t_i, N – количество снимков звездной камеры за рассматриваемый отрезок времени.

Сначала найдем выражение для $x^u(t_i)$, $y^u(t_i)$, зная начальные условия $x^u(t_0)$, $y^u(t_0)$, $\dot{x}^u(t_0)$, $\dot{y}^u(t_0)$ и значения ускорений a_k^X в инерциальном пространстве для каждого момента t_k прихода измерений с блока IMU. В общем виде для $x^u(t_i)$ и $\dot{x}^u(t_i)$ оно запишется следующим образом:

$$x^u(t_k) = x^u(t_{k-1}) + \dot{x}^u(t_k)\left(t_k - t_{k-1}\right) + \frac{a_k^X}{2}\left(t_k - t_{k-1}\right),$$

$$\dot{x}^u(t_k) = \dot{x}^u(t_{k-1}) + a_k^X\left(t_k - t_{k-1}\right).$$

Если же в этих формулах все выразить через начальные условия, ускорения a_k^X и время измерений, то получился следующая формула:

$$x^u(t_k) = x^u(t_0) + \dot{x}^u(t_0)\left(t_k - t_0\right) + \sum_{j=1}^{k-1}\sum_{l=j+1}^{k} a_j^X\left(t_j - t_{j-1}\right)\left(t_l - t_{l-1}\right) + \frac{3}{2}\sum_{j=1}^{k} a_j^X\left(t_j - t_{j-1}\right).$$

Теперь запишем, чему равны проекции ускорений в инерциальной системе отчета:

$$a_k^X = (a_k^x - \alpha)\cos(\varphi_k) + (a_k^y - \beta)\sin(\varphi_k),$$

$$a_k^Y = (a_k^y - \beta)\cos(\varphi_k) - (a_k^x - \alpha)\sin(\varphi_k).$$

Здесь a_k^x, a_k^y – измерения инклинометра в собственной системе отчета, α, β – смещения ноля по осям ox и oy соответственно, φ_i – угол поворота собственной системы отчета относительно инерциальной.

Теперь подставим выражения для ускорений в формулу для $x^u(t_k)$ и сгруппируем члены при α, β.

$$x^u(t_k) = x^u(t_0) + \dot{x}^u(t_0)\left(t_k - t_0\right) +$$

$$+\sum_{j=1}^{k-1}\sum_{l=j+1}^{k}\left[a_j^x\cos(\varphi_j) + a_j^y\sin(\varphi_j)\right]\left(t_j - t_{j-1}\right)\left(t_l - t_{l-1}\right) + \frac{3}{2}\sum_{j=1}^{k}\left[a_j^x\cos(\varphi_j) + a_j^y\sin(\varphi_j)\right]\left(t_j - t_{j-1}\right) +$$

$$+\alpha\left[\sum_{j=1}^{k-1}\sum_{l=j+1}^{k} -\cos(\varphi_j)\left(t_j - t_{j-1}\right)\left(t_l - t_{l-1}\right) + \frac{3}{2}\sum_{j=1}^{k} -\cos(\varphi_j)\left(t_j - t_{j-1}\right)\right] +$$

$$+\beta\left[\sum_{j=1}^{k-1}\sum_{l=j+1}^{k} -\sin(\varphi_j)\left(t_j - t_{j-1}\right)\left(t_l - t_{l-1}\right) + \frac{3}{2}\sum_{j=1}^{k} -\sin(\varphi_j)\left(t_j - t_{j-1}\right)\right].$$

Примем следующие обозначения:

$$f_k^x = x^u(t_0) + \dot{x}^u(t_0)\left(t_k - t_0\right) +$$
$$+\sum_{j=1}^{k-1}\sum_{l=j+1}^{k}\left[a_j^x\cos(\varphi_j) + a_j^y\sin(\varphi_j)\right]\left(t_j - t_{j-1}\right)\left(t_l - t_{l-1}\right) + \frac{3}{2}\sum_{j=1}^{k}\left[a_j^x\cos(\varphi_j) + a_j^y\sin(\varphi_j)\right]\left(t_j - t_{j-1}\right),$$

$$a_k^x = \left[\sum_{j=1}^{k-1}\sum_{l=j+1}^{k} -\cos(\varphi_j)\left(t_j - t_{j-1}\right)\left(t_l - t_{l-1}\right) + \frac{3}{2}\sum_{j=1}^{k} -\cos(\varphi_j)\left(t_j - t_{j-1}\right)\right],$$
$$b_k^x = \left[\sum_{j=1}^{k-1}\sum_{l=j+1}^{k} -\sin(\varphi_j)\left(t_j - t_{j-1}\right)\left(t_l - t_{l-1}\right) + \frac{3}{2}\sum_{j=1}^{k} -\sin(\varphi_j)\left(t_j - t_{j-1}\right)\right],$$

тогда выражение можно записать следующим образом:

$$x^u(t_k) = f_k^x + \alpha a_k^x + \beta b_k^x.$$

Для $y^u(t_k)$ можно получить аналогичное выражение:

$$y^u(t_k) = f_k^y + \alpha a_k^y + \beta b_k^y.$$

Вернемся к функционалу (1) и подставим в него найденные значения для $x^u(t_i)$, $y^u(t_i)$:

$$\Phi = \sum_{i=1}^{N}\left[\left(f_{k(i)}^x + \alpha a_{k(i)}^x + \beta b_{k(i)}^x - x^{эв}(t_i)\right)^2 + \left(f_{k(i)}^y + \alpha a_{k(i)}^y + \beta b_{k(i)}^y - y^{эв}(t_i)\right)^2\right].$$

Здесь обозначение $k(i)$ означает, что k-ое значение коэффициентов соответствует времени t_i.

Теперь продифференцируем функционал по неизвестным α, β и приравняем частные производные нулю:

$$\frac{\partial\Phi}{\partial\alpha} = \sum_{i=1}^{N}\left[a_{k(i)}^x\left(f_{k(i)}^x + \alpha a_{k(i)}^x + \beta b_{k(i)}^x - x^{эв}(t_i)\right) + a_{k(i)}^y\left(f_{k(i)}^y + \alpha a_{k(i)}^y + \beta b_{k(i)}^y - y^{эв}(t_i)\right)\right] = 0,$$
$$\frac{\partial\Phi}{\partial\beta} = \sum_{i=1}^{N}\left[b_{k(i)}^x\left(f_{k(i)}^x + \alpha a_{k(i)}^x + \beta b_{k(i)}^x - x^{эв}(t_i)\right) + b_{k(i)}^y\left(f_{k(i)}^y + \alpha a_{k(i)}^y + \beta b_{k(i)}^y - y^{эв}(t_i)\right)\right] = 0.$$

Сгруппируем члены при α, β и получим

$$\alpha\sum_{i=1}^{N}\left[\left(a_{k(i)}^x\right)^2 + \left(a_{k(i)}^y\right)^2\right] + \beta\sum_{i=1}^{N}\left[\left(a_{k(i)}^x b_{k(i)}^x\right) + \left(a_{k(i)}^y b_{k(i)}^y\right)\right] =$$
$$= \sum_{i=1}^{N}\left[a_{k(i)}^x\left(f_{k(i)}^x - x^{эв}(t_i)\right) + a_{k(i)}^y\left(f_{k(i)}^y - y^{эв}(t_i)\right)\right],$$
$$\alpha\sum_{i=1}^{N}\left[\left(a_{k(i)}^x b_{k(i)}^x\right) + \left(a_{k(i)}^y b_{k(i)}^y\right)\right] + \beta\sum_{i=1}^{N}\left[\left(b_{k(i)}^x\right)^2 + \left(b_{k(i)}^y\right)^2\right] =$$
$$= \sum_{i=1}^{N}\left[b_{k(i)}^x\left(f_{k(i)}^x - x^{эв}(t_i)\right) + b_{k(i)}^y\left(f_{k(i)}^y - y^{эв}(t_i)\right)\right].$$

.

В результате имеем линейную систему уравнений относительно двух неизвестных α, β. Введём следующие обозначения:

$$a_{11} = \sum_{i=1}^{N}\left[\left(a^x_{k(i)}\right)^2 + \left(a^y_{k(i)}\right)^2\right],$$

$$a_{12}\sum_{i=1}^{N}\left[\left(a^x_{k(i)}\, b^x_{k(i)}\right) + \left(a^y_{k(i)} b^y_{k(i)}\right)\right],$$

$$a_{22} = \sum_{i=1}^{N}\left[\left(b^x_{k(i)}\right)^2 + \left(b^y_{k(i)}\right)^2\right],$$

$$f_1 = \sum_{i=1}^{N}\left[a^x_{k(i)}\left(f^x_{k(i)} - x^{зв}(t_i)\right) + a^y_{k(i)}\left(f^y_{k(i)} - y^{зв}(t_i)\right)\right],$$

$$f_2 = \sum_{i=1}^{N}\left[b^x_{k(i)}\left(f^x_{k(i)} - x^{зв}(t_i)\right) + b^y_{k(i)}\left(f^y_{k(i)} - y^{зв}(t_i)\right)\right].$$

Тогда систему можно записать следующим образом: $A\boldsymbol{\xi} = \mathbf{f}$,

где $\boldsymbol{\xi} = \begin{bmatrix}\alpha & \beta\end{bmatrix}$, $\mathbf{f} = \begin{bmatrix}f_1 & f_2\end{bmatrix}$, $A = \begin{bmatrix}a_{11} & a_{12}\\ a_{12} & a_{22}\end{bmatrix}$. Решение этой системы записывается следующим образом: $\boldsymbol{\xi} = A^{-1}\mathbf{f}$.

Теперь решим задачу нахождения смещения ноля описанным выше методом для эксперимента, рассмотренного в п.3.3.2. На рис. 18 изображены результаты такого поиска. Найденные значения смещения ноля $\alpha = 0{,}005 м/с^2$, $\beta = -0{,}004 м/с^2$ минимизируют ошибку определения положения двумя методами.

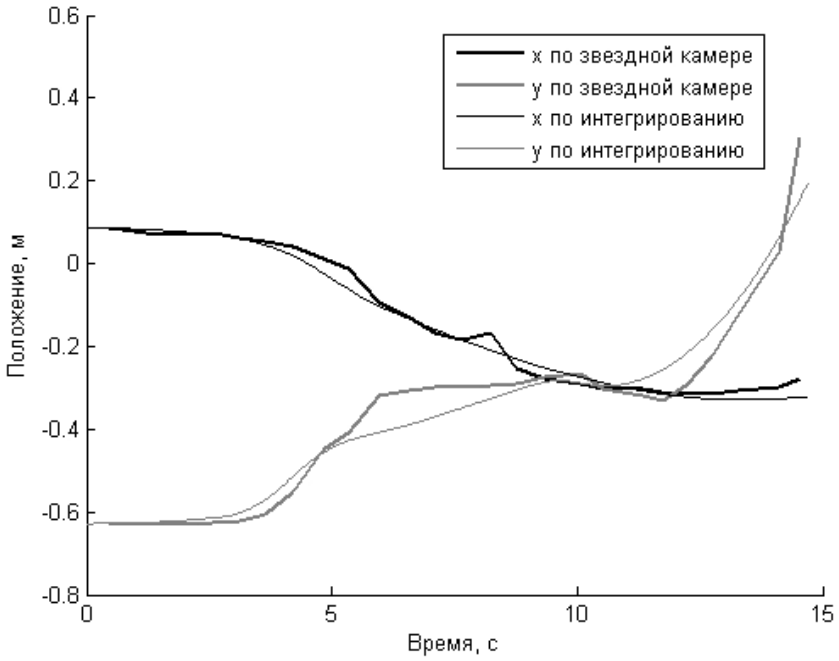

Рис. 18. Определение смещений ноля с помощью МНК

Однако, если рассмотреть более длительный эксперимент, то с помощью МНК уже не удаётся найти α, β, которые бы обеспечивали достаточное сов-

падение положения, полученные по звездной камере и по интегрированию. Это можно объяснить то, что смещение ноля хоть и слабо зависит от температуры ($\pm 0.0003 м / с^2 / ^\circ C$), но со временем изменяется, что приводит в конечном счете к дрейфу, и метод наименьших квадратов не может решить эту проблему.

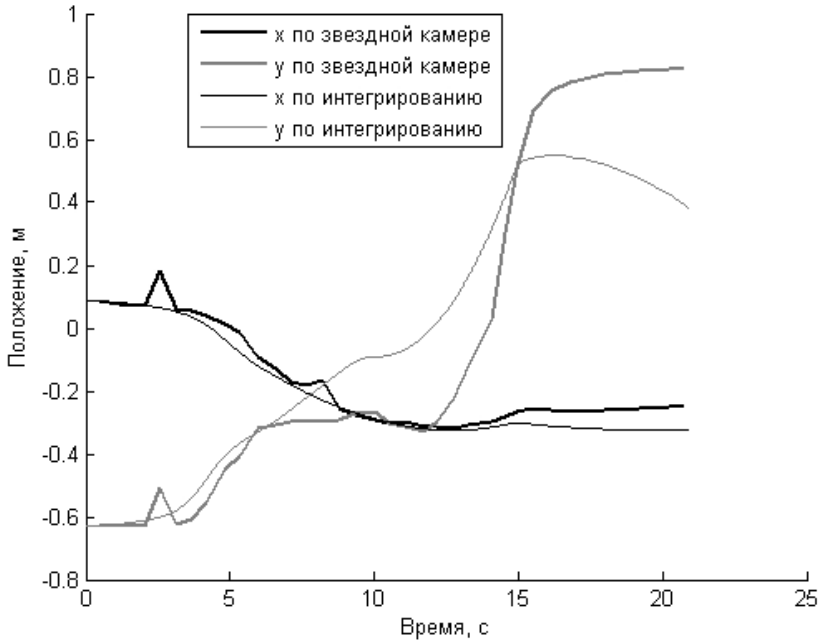

Рис. 19. Определение смещений ноля с помощью МНК на 20 с.

4. Алгоритм определения положения и ориентации с помощью инерциальных датчиков и звездной камеры

В предыдущих двух разделах настоящей работы были рассмотрены два независимых средства определения вектора состояния объекта и у каждого средства были свои плюсы и недостатки. Из основных недостатков использования звездной камеры можно назвать слишком малую частоту выдачи измерений, которая не позволяет непрерывно управлять макетом, и запаздывание измерений на время, примерно равное 0.6 с. При использовании инерциальных датчиков для получения вектора состояния основной проблемой является неточность знания смещения ноля инклинометров, что приводит к дрейфу. Поэтому построим алгоритм, который будет основан на обоих средствах измерения.

4.1. Описание алгоритма

Рассмотрим следующую схему алгоритма. Алгоритм начинает работать в тот момент времени, когда звездная камера делает первый снимок. Во время этого начинают интегрироваться данные акселерометра и угловой скорости с грубыми начальными условиями. При этом все измерения ускорений сохраняются в некоторый массив данных. В тот момент времени, когда вычисленный по снимку звездной камеры вектор состояния поступает по Wi-Fi на бортовой компьютер, производится повторное интегрирование ускорений с начальными условиями, равными вычисленным по снимку звездной камеры. Таким образом уточняется вектор состояния в настоящий момент времени, который продолжа-

ет вычисляться на основе интегрирования измерений ускорения. При получении начальных данных со звездной камеры проводится повторное интегрирование ускорений, сохраненных с момента получения последнего вектора состояния со звездной камеры.

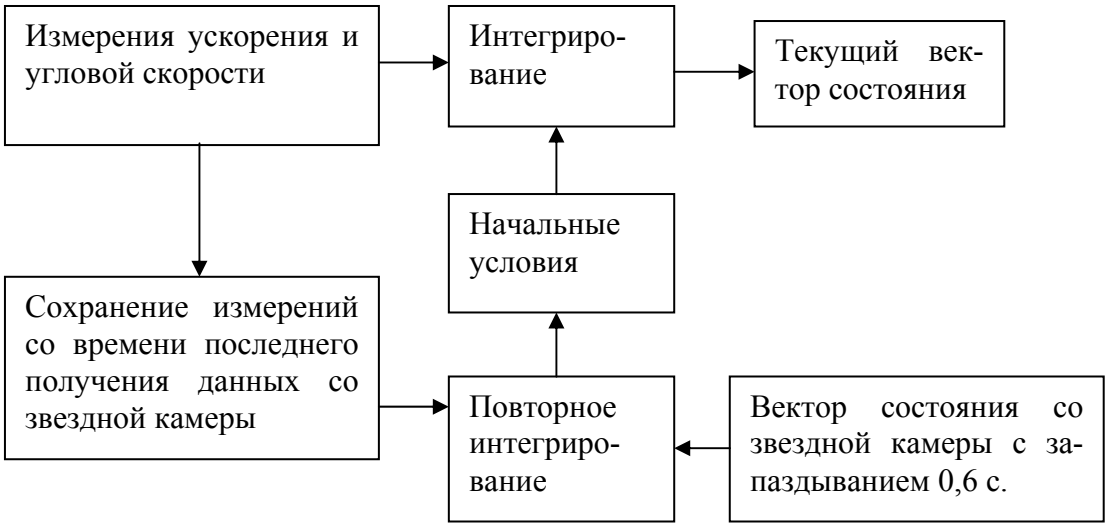

Рис. 20. Блок-схема алгоритма

Подобная схема лишена недостатков отдельных средств, так как частота выдачи вектора состояния равна частоте опроса инерциальных датчиков, и величина ухода текущего вектора состояния весьма мала, так как постоянно идет обновление начальных условий интегрирования, то есть ошибка за 0,6 секунд не успевает накопиться.

4.2. Результаты экспериментов

На рис. 21 и 22 представлены графики зависимости скорости и положения макета в инерциальной системе отчета, полученные с помощью предлагаемого алгоритма определения ориентации на основе двух независимых средств определения вектора состояния макета, а также жирными линиями обозначены скорости и координаты центра масс макета, полученные только на основе звездной камеры, а на рис. 23 и 24 дляудобства изображены графики разностей между ними. Как видно из этих графиков, данные, полученные по разработанному алгоритму, неплохо соответствуют данным, полученным со звездной камеры, и ошибка по положению в основном находится в пределах 5 см за исключением нескольких выбрасов. Таким образом, предложенный алгоритм решил проблему запаздывания данных, поступающих со звездной камеры, и проблему дрейфа, при интегрировании ускорений.

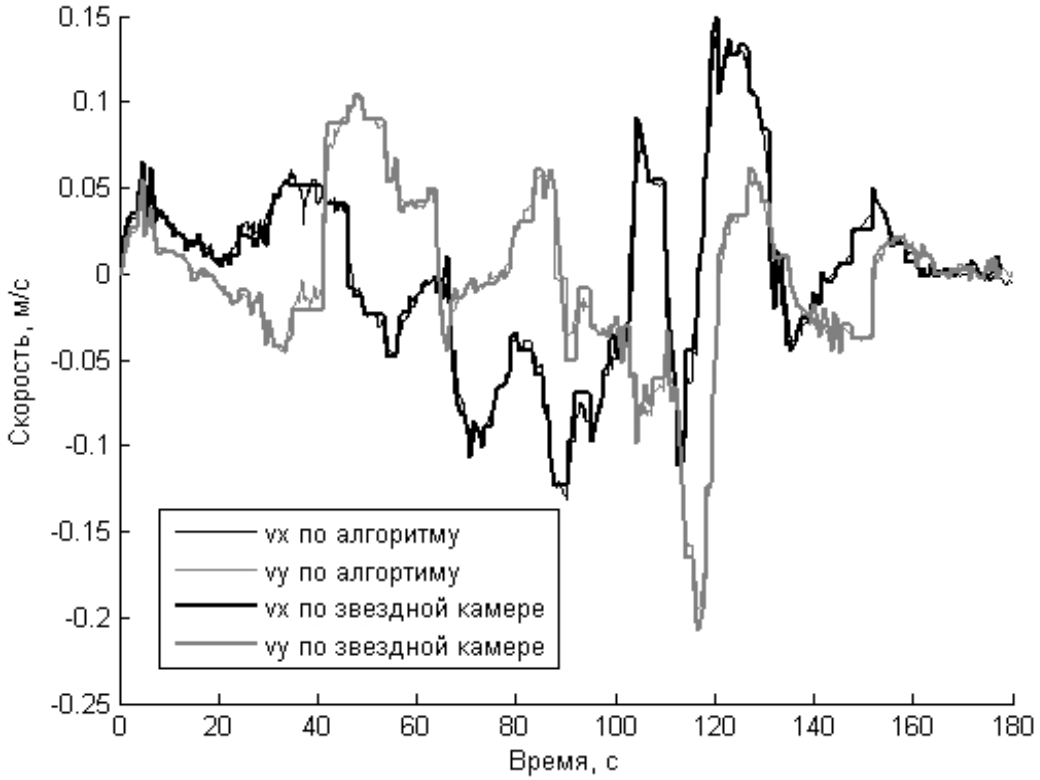

Рис. 21. Компоненты скорости макета в инерциальной системе координат

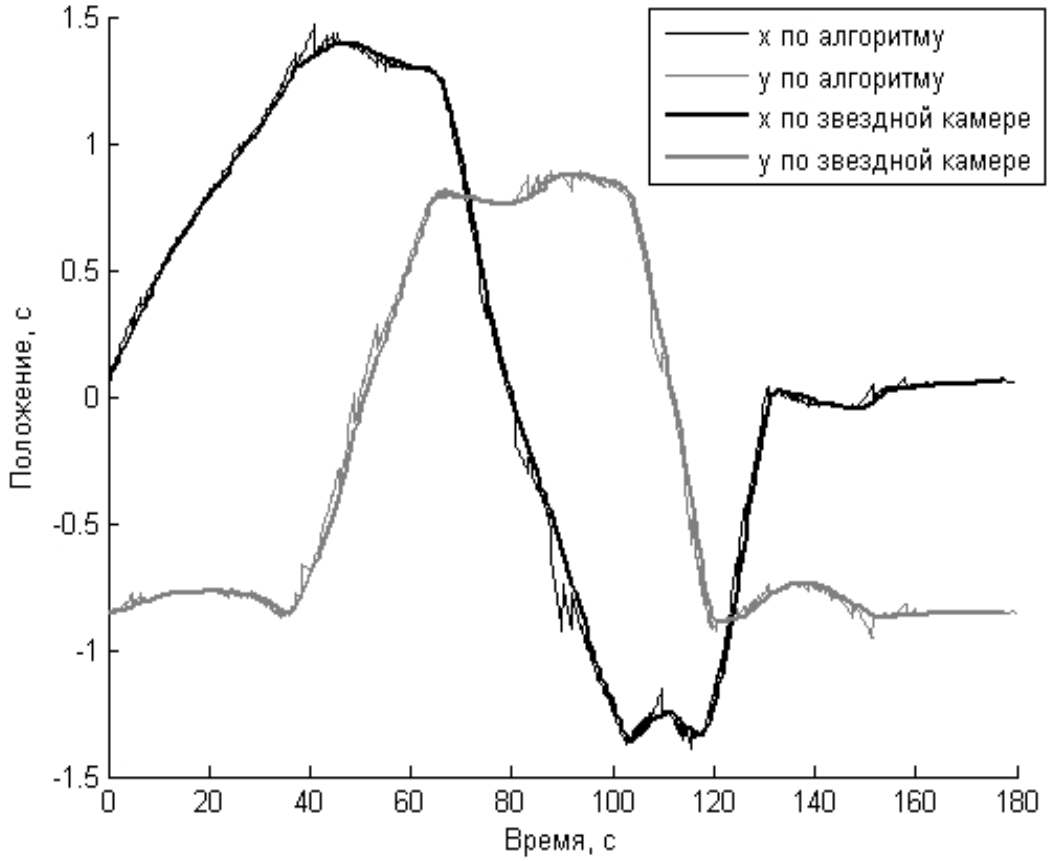

Рис. 22. Положение центра масс макета

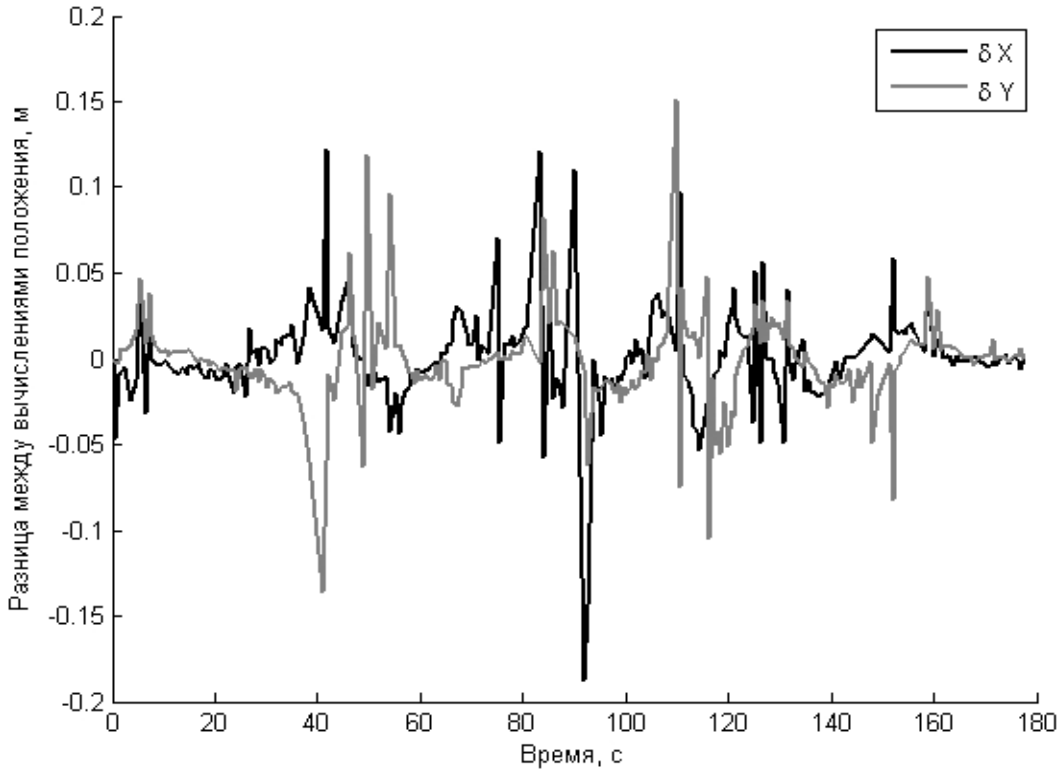

Рис. 23. Разность между положениями, вычисленными по предлагаемому алгоритму и по алгоритму, использующему только звездную камеру

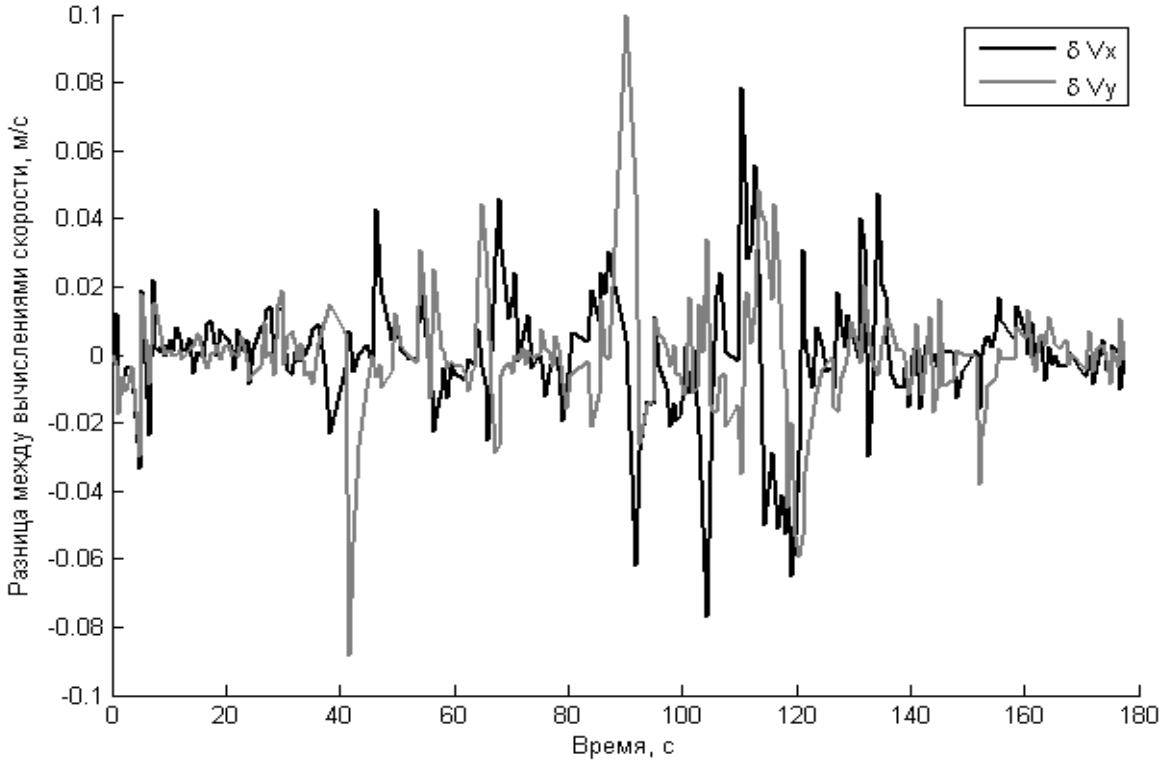

Рис. 24. Разность между скоростями, вычисленными по предлагаемому алгоритму и по алгоритму, использующему только звездную камеру

Заключение

Подробно рассмотрены и исследованы два независимых средства определения вектора состояния макета спутника, движущегося по горизонтальной поверхности на плоском столе. Это − звездная камера и блок инерциальных датчиков. Было показано, что каждое из них обладает определёнными недостатками, которые удается устранить при одновременном использовании этих средств. В дальнейшей работе авторы планируют использовать полученные результаты при управлении макетами спутников для отработки алгоритмов управления и навигации группы спутников.

Благодарности

Работа выполнена при поддержке DAAD, РФФИ (грант № 09-01-00431) и Минобранауки (госконтракт № 02.740.11.0860).

Список литературы

1. Д.С.Иванов, Т.Вальтер, Д.Биндель, М.Ю.Овчинников. Стенд для отработки алгоритмов управления движением многоэлементных систем //Препринт ИПМ им. М.В.Келдыша РАН, Москва, 2008 г., №56, 32с.

2. Д.Биндель, И.Е.Зараменских, Д.С.Иванов, М.Ю.Овчинников, Н.Г.Прончева. Лабораторный стенд для верификации алгоритмов управления группировкой спутников // Известия РАН. Теория и системы управления. 2009, №5, с. 109-117.

3. Д.С.Иванов, М.Ю.Овчинников. Математическое моделирование управляемого движения многоэлементной системы //Препринт ИПМ им. М.В.Келдыша РАН, Москва, 2008 г., №78, 32с.

4. Д.С.Иванов, М.Ю.Овчинников, С.П.Трофимов. Применение фотограмметрического метода в задаче автономного определения относительного движения группы макетов //Препринт ИПМ им. М.В.Келдыша РАН, Москва, 2010 г., №5, 30с.

5. *Z.L.Wang, W.Quan*. An All-Sky Autonomous Star Map Identification Algorithm. IEEE Aerospace and Electronic Systems Magazine, No. 19, pp. 10-14.

6. *M.A.Samaan, D.Mortari, J.L.Junkins*. Non-Dimensional Star Identification for Uncalibrated Star Cameras. AAS/AIAA Space Flight Mechanics Meeting, Ponce, Puerto Rico, Paper No. AAS 03-131, February, 2003.

7. М.Грасси, А.А.Дегтярев, М.Ю.Овчинников, А.Перротта. Тестирование и калибровка прототипа миниатюрного солнечного датчика, разработанного на основе APS-технологии //Препринт ИПМ им. М.В.Келдыша РАН, Москва, 2005 г., №94, 30с.

Содержание

1. Введение 3
2. Алгоритм идентификации участка звездного неба 4
2.1. Классификация алгоритмов 4
2.2. Особенность разработанного алгортма 5
2.3. Описание алгоритма 7
2.3.1. Нахождение координат центров звёзд 7
2.3.2. Структура базы данных 7
2.3.3. Извлечение информации об участке неба на снимке 8
2.3.4. Поиск по базе данных (каталогу) 8
2.3.5. Проверка на ошибку (валидация) 9
2.4. Реализация алгоритма 9
2.4.1. Быстродействие 11
2.4.2. Точность 13
3. Алгоритм определения положения и ориентации с помощью инерциальных датчиков 13
3.1. Недостатки использования инерциальных датчиков 14
3.2. Алгоритм определения вектора состояния 14
3.2.1. Предварительная калибровка 15
3.2.2. Интегрирование измерений 15
3.3. Реализация алгоритма 16
3.3.1. Описание блока датчиков 16
3.3.2. Результаты экспериментов 17
3.3.3. Нахождение смещения ноля инклинометра 20
4. Алгоритм определения положения и ориентации с помощью инерциальных датчиков и звездной камеры 24
4.1. Описание алгоритма 24
4.2. Результаты экспериментов 25
Заключение 28
Список литературы 29