УДК 528.91
DOI 10.33764/2618-981X-2025-1-1-128-138

В. Е. Терещенко[1]✉, В. А. Машинец[1], Э. Э. Банщикова[1]

Определение координат недоступной точки с помощью тахеометра и отражателя методом аппроксимации шара

[1]Сибирский государственный университет геосистем и технологий, г. Новосибирск, Российская Федерация
e-mail: v.e.tereshenko@sgugit.ru

Аннотация. В статье представляется метод определения координат точки с помощью электронного тахеометра и геодезического отражателя на вехе. При этом веха на точке располагается не в вертикальном положении, как это требуется при классической передаче координат с точки стояния тахеометра, а в наклонном положении, которое вызвано отсутствием прямой видимости вследствие наличия препятствий между тахеометром и отражателем. Наклоны вехи с отражателем с фиксацией ее острия на определяемой точке формируют поверхность полусферы. При наличии измерений до нескольких точек, описывающих поверхность полусферы, можно составить уравнение сферы, а затем математически вычислить координаты ее цента. Они будут являться координатами определяемой точки, прямая видимость до которой ограничена препятствием. В результате исследования сформирована программа, которая принимает в качестве исходных данных координаты точек на поверхности полусферы, проводит обработку и, если поступившие данные имеют удовлетворительную геометрию, происходит вычисление координат точки. Данный метод определения координат реализован с помощью современного языка программирования Python, с помощью которого вычислительный процесс максимально автоматизирован. В статье приведены примеры работы программы на реальных измерениях. Отклонения от истинных значений не превышают погрешность центрирования прибора в совокупности с погрешностью удержания вехи в наклонном положении над точкой.

Ключевые слова: электронный тахеометр, призменный отражатель, аппроксимация шара, Python, геодезия

V. E. Tereshchenko[1]✉, V. A. Mashinec[1], E. E. Banshchikova[1]

Coordinates calculating of an unobtainable point with a tacheometer-reflector system using the sphere approximation method

[1] Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
e-mail: v.e.tereshenko@sgugit.ru

Abstract. The article provides an overview of the method for determining the coordinates of a point using an electronic tacheometer and a geodetic reflector on a pole. In this case, the pole on the point is not located in a vertical position, as is required for the classical transfer of coordinates from the point where the tacheometer is located, but in an inclined position, which is caused by the impossibility of maintaining direct visibility due to the presence of obstacles between the tacheometer and the reflector. The tilts of the pole with the reflector with its tip fixed on the point being determined form the surface of a hemisphere. If there are measurements of several points describing the surface of the hemisphere, it is possible to create an equation of the sphere, and then mathematically calculate the coordinates of its center. They will be the coordinates of the point being determined, the direct visibility to which is limited by an obstacle. As a result of the study, a program was created that accepts the coordinates of points on the surface of the hemisphere as initial data, performs processing

and, if the received data have satisfactory geometry, the coordinates of the point are calculated. This method of determining coordinates is implemented using the modern Python programming language, with the help of which the computational process is maximally automated. The article provides examples of the program's operation on real measurements. Deviations from the true values do not exceed the error in centering the device together with the error in holding the pole in an inclined position above the point.

Keywords: electronic tacheometer, prism reflector, sphere approximation, Python, geodesy

Введение

В геодезии, строительстве и мониторинге инженерных сооружений часто возникают ситуации, когда необходимо определить координаты точки, которая физически недоступна для прямых измерений тахеометром (например, точка закрыта стволом дерева, столбом или иным препятствием). Методы определения координат с помощью технологий ГНСС могут быть не доступны в закрытых помещениях или в условиях плотной застройки. Также может быть ключевым вопрос точности в сравнении с классическими методами «тахеометр-призма» [1, 2]. Именно для таких случаев может применяться метод аппроксимации шара. Вследствие приведенных выше факторов исследования в этом направлении выглядят актуальными.

Целью данного исследования является создание компьютерной программы для определения координат недоступной точки методом аппроксимации шара. Программа может быть загружена в тахеометры как одно из прикладных приложений для облегчения процесса производства геодезических работ. В соответствии с целью исследования необходимо решить следующие задачи:

– смоделировать данные в виртуальной среде для отладки алгоритмов вычисления;

– создать алгоритм вычисления центра сферы по известным точкам на ее поверхности;

– провести натурные испытания алгоритма на реальных измерениях с помощью тахеометра Leica TCR405 и комплектного отражателя;

– провести оценку точности полученных координат с помощью созданной программы, сравнив вычисленные результаты с эталонными;

– сделать выводы об изменении точности результатов вычислений при включении дополнительных данных, улучающих геометрию.

Аппроксимация сферы

Аппроксимация шара – это математический метод определения центра сферы по набору точек, лежащих на её поверхности. Суть метода заключается в следующем:

– отражатель перемещается по поверхности воображаемой сферы с центром в недоступной точке, при этом радиус сферы равен высоте отражателя, а центр сферы находится ровно в точке определяемого пункта;

– электронным тахеометром измеряются направления и расстояния до отражателя в разных позициях, доступных при непосредственной видимости;

– сфера аппроксимируется, и её центр (искомая точка) вычисляется математически.

На рисунке 1 представлена схема, представляющая метод аппроксимации шара в плановом (плоском, вид сверху) положении.

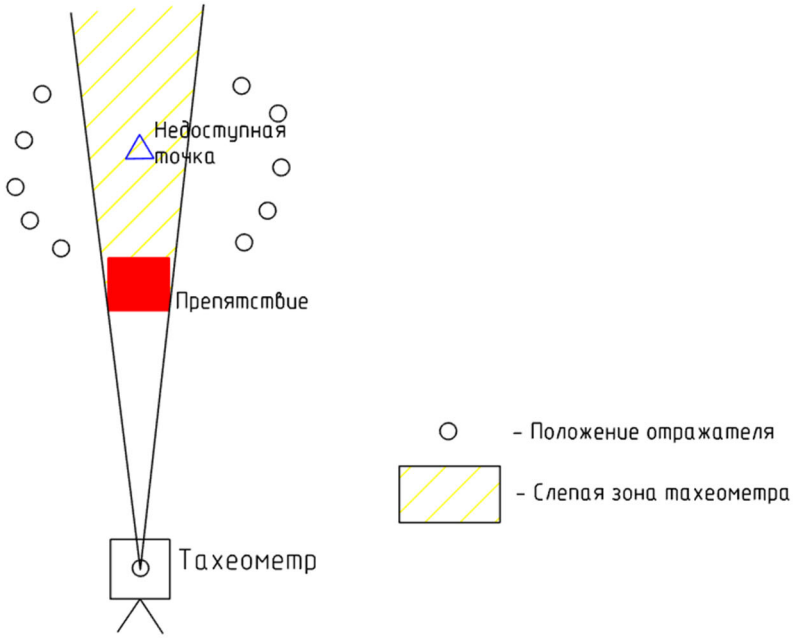


Рис. 1. Метод аппроксимации шара

Для составления программы использовался язык программирования Python и виртуальная среда разработки Visual Studio Code. Формулы для математических расчетов координат центра сферы по точкам на ее поверхности получены с помощью нейросети Deepseek. В промте дополнительно указывалось требование о выводе источников данных, откуда нейросеть брала информацию. В выводе результатов нейросеть указала работы авторов: Pratt V., Gander W., Lukács G., Ahn S., Hesse R., все на тему «подбор сфер методом наименьших квадратов» ссылки на источники приведены в [3-7]. Формулы из этих работ использованы в алгоритме программы:

$$a = \frac{1}{D} \cdot \begin{vmatrix} A & y_2 - y_1 & z_2 - z_1 \\ B & y_3 - y_1 & z_3 - z_1 \\ C & y_4 - y_1 & z_4 - z_1 \end{vmatrix},$$

$$b = \frac{1}{D} \cdot \begin{vmatrix} x_2 - x_1 & A & z_2 - z_1 \\ x_3 - x_1 & B & z_3 - z_1 \\ x_4 - x_1 & C & z_4 - z_1 \end{vmatrix},$$

$$c = \frac{1}{D} \cdot \begin{vmatrix} x_2 - x_1 & y_2 - y_1 & A \\ x_3 - x_1 & y_3 - y_1 & B \\ x_4 - x_1 & y_4 - y_1 & C \end{vmatrix},$$

где

$$D = (x_2 - x_1)(y_3 - y_1)(z_4 - z_1) + (y_2 - y_1)(z_3 - z_1)(x_4 - x_1) + (z_2 - z_1)(x_3 - x_1)(y_4 - y_1) - (z_2 - z_1)(y_3 - y_1)(x_4 - x_1)(z_4 - z_1) - (x_2 - x_1)(z_3 - z_1)(y_4 - y_1),$$

а коэффициенты *A*, *B*, *C*:

$$A = \frac{x_2^2 - x_1^2 + y_2^2 - y_1^2 + z_2^2 - z_1^2}{2},$$

$$B = \frac{x_3^2 - x_1^2 + y_3^2 - y_1^2 + z_3^2 - z_1^2}{2},$$

$$C = \frac{x_4^2 - x_1^2 + y_4^2 - y_1^2 + z_4^2 - z_1^2}{2}.$$

при этом *x1, y1, z1, x2, y2, z2, x3, y3, z3, x4, y4, z4* – координаты точек на поверхности аппроксимируемой сферы.

Моделирование данных в виртуальной среде

Для корректной работы программы, отладки алгоритмов, поиска багов и ошибок вычислений, смоделирован набор точек, лежащих на поверхности сферы, который имитировал результаты реальных измерений, но в виртуальной среде. Он получен с помощью сайта https://www.geogebra.org (рисунок 2). В виртуальной среде создана сфера радиусом 1.5 метра (приближенная к реальной высоте отражателя) и центром в начале координат. С помощью соответствующих инструментов на сайте расставлены 8 точек на поверхности сферы по 4 точки слева и справа от места наблюдения. Вблизи середины сферы точки не моделированы, так как в реальных условиях они скрыты за препятствием.

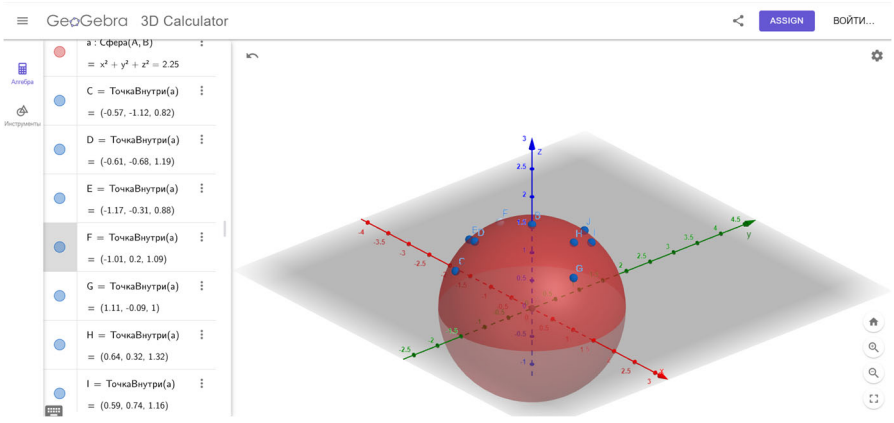


Рис. 2. Интерфейс сайта GeoGebra

На рисунке 3 приведен пример координат всех точек, собранных в текстовый файл в следующем формате: каждая строка соответствует номеру точки, в столбцах координаты X, Y и Z соответственно. Разделитель между целой и деся-

тичной частью числа – точка. Разделителем между столбцами служит запятая с пробелом. В итоге именно в таком формате программа принимает измерения для вычисления координат недоступной точки.

Здесь и далее координатой Z является высотная компонента при определении центра шара. Она эквивалентна общепринятому обозначению высоты H, направлена в зенит, совпадает с отвесной линией в точке.

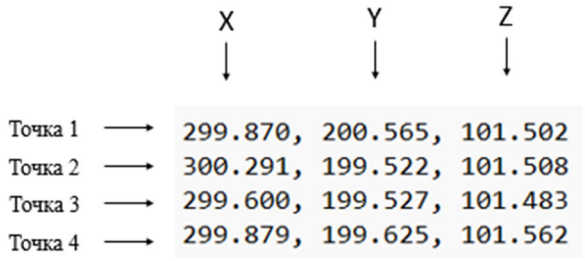
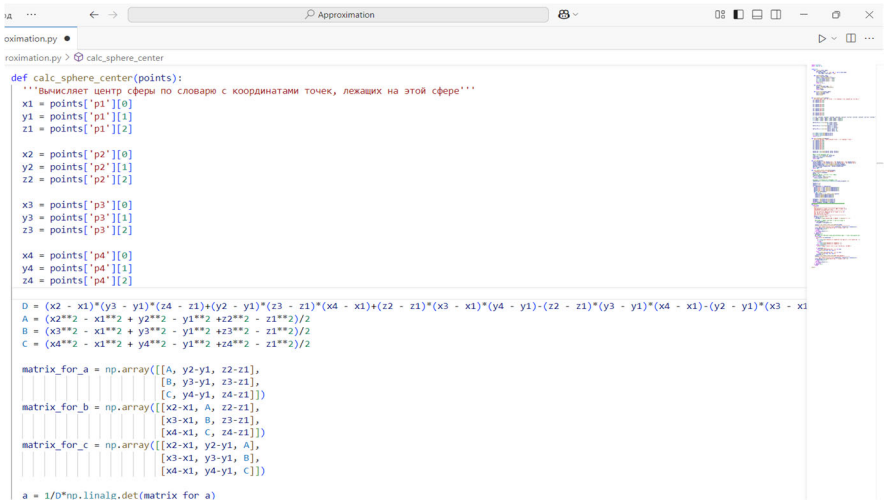


Рисунок 3 – Формат записи данных в текстовый файл

Алгоритм программы

При реализации функции для аппроксимации шара использовалась библиотека Python – numpy, так как в ней реализованы инструменты для работы с матрицами в Python. Часть кода, содержащая функцию аппроксимации шара, представлена на рисунке 4.



```
def calc_sphere_center(points):
    '''Вычисляет центр сферы по словарю с координатами точек, лежащих на этой сфере'''
    x1 = points['p1'][0]
    y1 = points['p1'][1]
    z1 = points['p1'][2]

    x2 = points['p2'][0]
    y2 = points['p2'][1]
    z2 = points['p2'][2]

    x3 = points['p3'][0]
    y3 = points['p3'][1]
    z3 = points['p3'][2]

    x4 = points['p4'][0]
    y4 = points['p4'][1]
    z4 = points['p4'][2]

    D = (x2 - x1)*(y3 - y1)*(z4 - z1)+(y2 - y1)*(z3 - z1)*(x4 - x1)+(z2 - z1)*(x3 - x1)*(y4 - y1)-(z2 - z1)*(y3 - y1)*(x4 - x1)-(y2 - y1)*(x3 - x1
    A = (x2**2 - x1**2 + y2**2 - y1**2 +z2**2 - z1**2)/2
    B = (x3**2 - x1**2 + y3**2 - y1**2 +z3**2 - z1**2)/2
    C = (x4**2 - x1**2 + y4**2 - y1**2 +z4**2 - z1**2)/2

    matrix_for_a = np.array([[A, y2-y1, z2-z1],
                             [B, y3-y1, z3-z1],
                             [C, y4-y1, z4-z1]])
    matrix_for_b = np.array([[x2-x1, A, z2-z1],
                             [x3-x1, B, z3-z1],
                             [x4-x1, C, z4-z1]])
    matrix_for_c = np.array([[x2-x1, y2-y1, A],
                             [x3-x1, y3-y1, B],
                             [x4-x1, y4-y1, C]])

    a = 1/D*np.linalg.det(matrix_for_a)
```

Рис. 4. Фрагмент скриншота кода аппроксимации шара

Для обработки избыточных измерений (более 4 исходных точек), использована библиотека Python – itertools. Она позволяет создавать все возможные уникальные комбинации любого количества точек при обработке.

Алгоритм программы учитывает пространственное положение введенных точек и сразу отбрасывает комбинации, имеющие неудовлетворительную гео-

метрию. На рисунке 5 представлена блок-схема работы алгоритма отбраковки данных, представляющих не оптимальную геометрию.

Для оценки геометрии в программу добавлен коэффициент понижения точности DOP (с англ. Dilution of Precision – снижение точности), аналогичный тому, что используется при оценке геометрического фактора спутников ГНСС. В данном случае оценка коэффициента DOP упрощена, но смысл тот же. Коэффициент DOP обратно пропорционален площади четырехугольника, образованным любыми положениями отражателя в четырех точках. В ходе многократных изменений коэффициента установлено его предельное значение для положительного решения о включении четырехугольника в обработку. Он равен 2,5. На рисунке 6 приведены положения точек для вычисления по ним функции шара с «плохой» и «хорошей» геометрией.

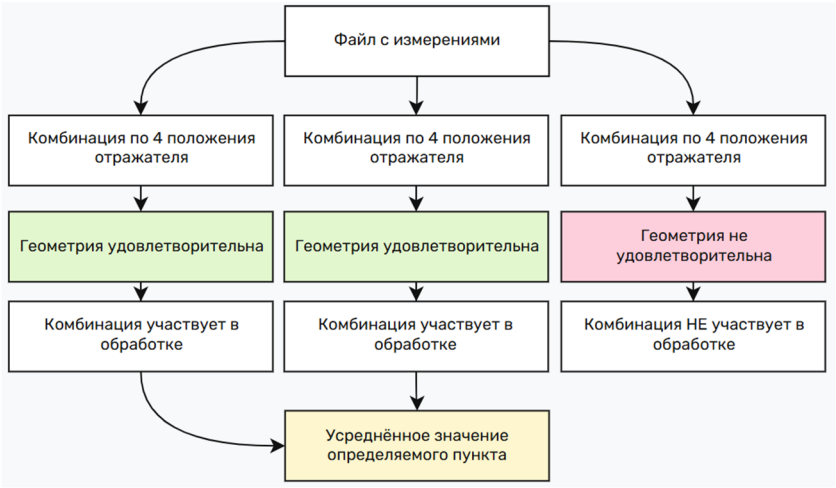


Рис. 5. Блок-схема алгоритма

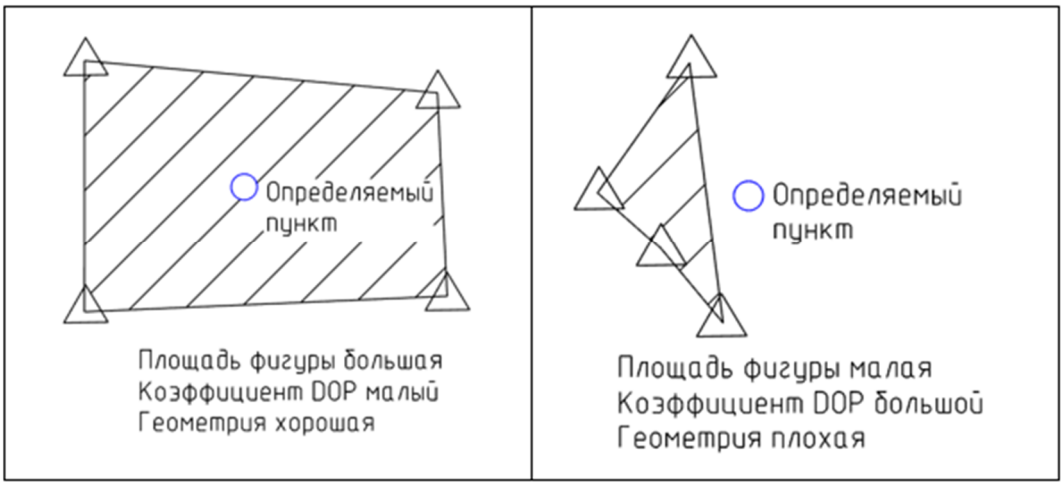


Рис. 6. Коэффициент DOP

При предварительном тестировании программы вертикальная координата Z вычислялась с относительно большой погрешностью – порядка нескольких сан-

тиметров. Такой результат был связан с плохой геометрией пунктов в вертикальной плоскости. Исключить эту проблему можно добавлением в программу функции для уточнения координаты Z. Для этого добавлено дополнительное поле ввода высоты отражателя, с которой проводилась съемка.

Теперь алгоритм работы программы выглядит иначе. После вычисления центра сферы заново перебираются все комбинации точек, вычисляются расстояния между полученным центром и положением зеркала отражателя, затем расстояния усредняются. Таком образом для каждой комбинации из четырех положений отражателя находится псевдо-длина отражателя. Находится разность между псевдо-длиной отражателя и действительной высотой отражателя, указанной пользователем. Далее эта разность учитывается как поправка в координату Z. Используя такой подход удалось повысить точность определения вертикальной компоненты в несколько раз.

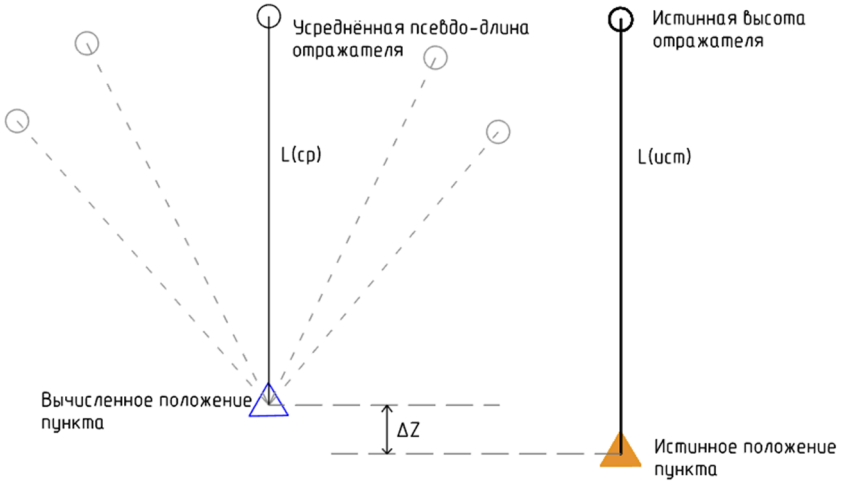


Рис. 7. Схема уточнения высотной компоненты

Для создания пользовательского интерфейса использовалась библиотека Python – tkinter. В результате создана программа в виде исполняемого файла с привычным интерфейсом. На рисунке 8 приведен скриншот запущенной программы.

Программа запускается и работает на любом ПК без установки Python. Файл с измерениями выбирается через встроенный проводник Windows после нажатия на кнопку «Открыть файл». Если известна высота отражателя ее можно указать в нижней части окна программы, после нажать «Установить высоту». После того, как все необходимые настройки указаны, кликнув кнопку «Вычислить центр» будет получен результат вычислений. Дополнительно в программу добавлен ручной ввод координат с возможностью сохранить их в файл поддерживаемого формата.

Добавлена функция интерактивной визуализации положения призмы отражателя и вычисленного пункта для анализа их взаимной геометрии. Для того чтобы открыть окно визуализации необходимо сначала провести вычисления в

программе, после нажать на кнопку «Визуализация». На рисунке 9 приведен фрагмент окна «Визуализация» программы.

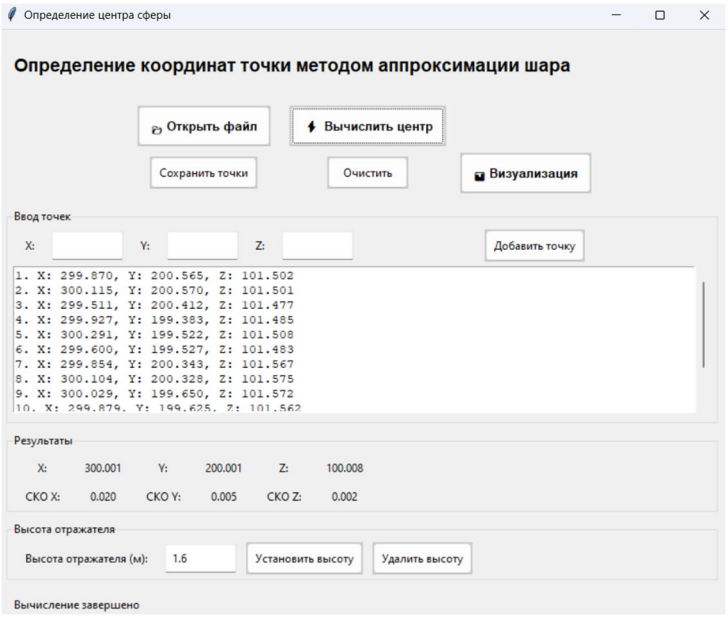


Рис. 8. Главное окно программы

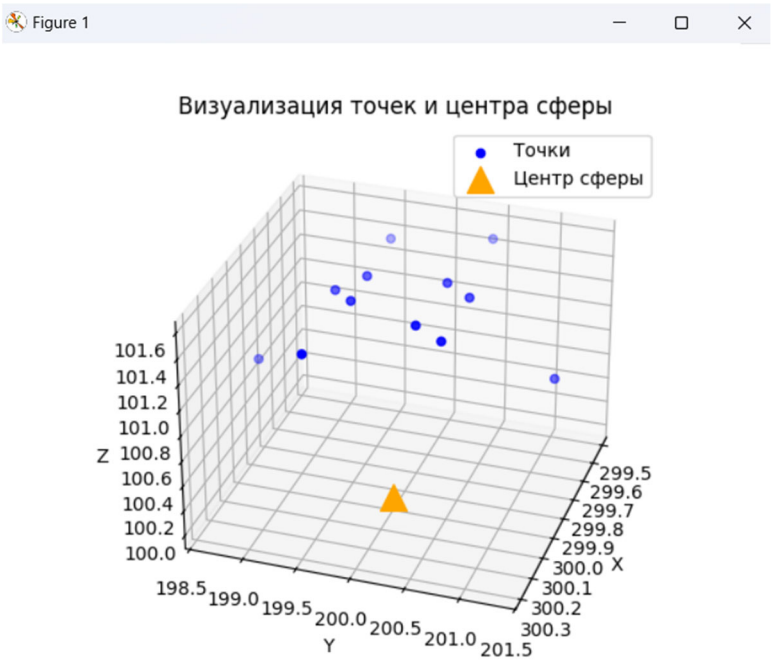


Рис. 9. Окно «Визуализация»

Важно указать, что при определении координат способом аппроксимации шара необходимо соблюсти следующие требования: в параметрах съемки тахеометра необходимо указать высоту отражателя равной нулю для того, чтобы получать координаты непосредственно самого зеркала отражателя, высота отражателя не должна изменяться в процессе съемки, конец вешки должен находится в определяемом пункте.

Апробация программно-вычислительного комплекса

Работа программы апробирована на реальных измерениях. Оценка точности выполнялась путем сравнения истинных координат и вычисленных в сформированной программе. Истинные координаты определены тахеометром при закреплении условной системы координат. Вычисленные координаты определяемой точки получены основе реальных измерений с помощью тахеометра Leica TCR405 и комплектного отражателя на вехе. Тахеометр на штативе был установлен в аудитории Сибирского государственного университета геосистем и технологий (СГУГиТ) на точке ТТ-1. В качестве недоступной точки выбрана точка ТТ-5 с координатами 300.000, 200.000 и 100.000 метров X, Y и Z соответственно.

Определено 10 положений отражателя: 5 слева и 5 справа от воображаемого препятствия. В эксперименте приняты 2 положения отражателя с большим углом наклона вехи и 8 измерений с малым, как показано на рисунке 10 (*а*, *б*).

а)

б)

Рис. 10. Экспериментальные измерения

В таблице представлены результаты измерений с последующей их обработкой в созданной программе, а также разницы вычисленных координат с эталонными значениями ΔX, ΔY, ΔZ.

Вычисленные координаты центра сферы
и их разницы с эталонными значениями

Измерения	X	Y	Z	ΔX	ΔY	ΔZ
Эталонные координаты	300.000	200.000	100.000	–	–	–
Измерения с малым наклоном вехи	299.999	200.002	100.041	0.001	0.002	0.041
Измерения с большим наклоном вехи	300.001	200.001	100.033	0.001	0.001	0.033
Измерения с малым наклоном вехи с учетом длины отражателя	299.999	200.002	100.010	0.001	0.002	0.010
Измерения с большим наклоном вехи с учетом длины отражателя	300.001	200.001	100.008	0.001	0.001	0.008

В таблице видно, что измерения с большим углом наклона вехи дают лучшее определение вертикальной координаты Z, при этом существенно не влияя на определение плановых координат. Также очевидно, что введение высоты отражателя в исходные данные положительно влияет на точность определения координаты Z. При наличии высоты отражателя отклонение по высотной компоненте уменьшается почти в 4 раза и составляет 8 мм.

Заключение

В результате проведенного исследования и разработки программы для аппроксимации шара для вычисления координат недоступной точки, сделаны следующие выводы:

– смоделированы данные в виртуальной среде на веб-сервисе;

– создана программа вычисления центра сферы по известным точкам на ее поверхности на языке Python;

– создан удобный пользовательский интерфейс программы;

– проведены натурные испытания алгоритма на реальных измерениях с помощью тахеометра Leica TCR405 и комплектного отражателя;

– проведена оценка точности полученных координат с помощью созданной программы, сравнив вычисленные результаты с эталонными. Экспериментальный вариант с большим наклоном вехи и с известной высотой отражателя оказался наиболее точным.

Исходя из решенных задач можно сделать вывод, что главная цель исследования – создание программы для определения координат недоступной точки методом аппроксимации шара достигнута. Отклонения от эталонных координат не превышают нескольких миллиметров.

Разработанный алгоритм нуждается в дальнейших исследованиях, например, на больших расстояниях. Или при вычислении точки, скрытой за углом здания, из-за чего в зоне видимости может быть только половина полусферы.

Код программы доступен по ссылке: https://clck.ru/3MKE8w. Любой может совершенствовать алгоритм или определить его слабые стороны.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Уставич, Г.А. Геодезия. В 2-х кн. Кн. 1 [Текст]: учебник для вузов / Г.А. Уставич. – Новосибирск: СГГА, 2012. – 352 с. ISBN 978-5-87693-487-1

2. Антонович, К.М. Использование спутниковых радионавигационных систем в геодезии [Текст]. В 2 т. Т. 1. Монография / К.М. Антонович; ГОУ ВПО «Сибирская государственная геодезическая академия». – М.: ФГУП «Картгеоцентр», 2005. – 334 с.: ил. ISBN 5-86066-071-5

3. Pratt V. Direct Least-Squares Fitting of Algebraic Surfaces [Текст] / V. Pratt // Computer Graphics. – 1987. – Vol. 21, no. 4. – P. 145–152. – DOI: 10.1145/37402.37422

4. Gander W. Least-Squares Fitting of Circles and Ellipses [Текст] / W. Gander, G. H. Golub, R. Strebel // BIT Numerical Mathematics. – 1994. – Vol. 34, no. 4. – P. 558–578. – DOI: 10.1007/BF01934268

5. Lukács G. Faithful Least-Squares Fitting of Spheres [Текст] / G. Lukács, A. D. Marshall, R. R. Martin // European Conference on Computer Vision (ECCV). – 1998. – P. 671–686. – DOI: 10.1007/BFb0055692

6. Ahn S. J. Least-Squares Orthogonal Distances Fitting [Текст] / S. J. Ahn // Geometric Fitting. – Berlin: Springer, 2004. – Chap. 3. – P. 41–60. – ISBN 978-3-540-20171-7.

7. Hesse R. A Fast Algorithm for Spherical Fitting [Текст] / R. Hesse, P. Leopardi // ANZIAM Journal. – 2007. – Vol. 48. – P. C187–C200. – DOI: 10.21914/anziamj.v48i0.70.

© В. Е. Терещенко, В. А. Машинец, Э. Э. Банщикова, 2025