УДК 004.5:004.9

Бєлозьоров Ж. О.,
викладач кафедри комп'ютерної інженерії,
ЧНУ ім. Петра Могили,
м. Миколаїв, Україна,
gan@mksat.net

АНАЛІЗ ТА РЕАЛІЗАЦІЯ АЛГОРИТМУ ОБЧИСЛЕННЯ КООРДИНАТ ПОСТРІЛУ НА БАЗІ МОБІЛЬНОГО ПРИСТРОЮ У ВЗАЄМОДІЇ З БПЛА

У статті вивчається можливість створення системи визначення координат джерела звуку (пострілу) на базі мобільного пристрою у взаємодії з БПЛА. Запропоновано і реалізовано метод градієнтного спуску з дробленням кроку в якості алгоритму для побудови мобільного додатку під Android. За допомогою розробленого прототипу програми перевірена працездатність цього алгоритму на смартфоні. Експериментальним шляхом досліджено вплив схеми розстановки мікрофонів на БПЛА на швидкість визначення координат пострілу і шляхи підвищення точності вихідних даних.

Ключові слова: *БПЛА; координата; джерело звуку; мікрофон; смартфон.*

Постановка проблеми. В попередній статті, в розрізі подій що розгортаються на полі бою, розглядається ситуація коли противник веде вогонь з декількох видів зброї, і завдання командира відділення полягає в тому, щоб проаналізувати всі вогневі точки супротивника, їх місце розташування до того як противник знищить наші. В даній ситуації на перше місце виходить фактор швидкості в оцінці обстановки та прийнятті рішення. А якщо розвідувальна інформація за допомогою БПЛА надходить до мобільного пристрою та після обробки потрапляє на екран у вигляді координат вогневих точок противника, із зазначенням типу ворожої зброї, та ще і в режимі реального часу?

Аналіз останніх досліджень і публікацій.

У сучасних задачах оптимізації прикладного характеру, цільова функція залежить від багатьох стартових параметрів. Мінімум диференційованої функції з великою кількістю змінних $U = f(x_1, x_2, ..., x_n)$ можна знайти, досліджуючи її значення в критичних точках (екстремумах), які визначаються рішенням системи диференціальних рівнянь

$$\frac{\partial f}{\partial x_1} = 0, \quad \frac{\partial f}{\partial x_2} = 0, \quad ..... \quad \frac{\partial f}{\partial x_n} = 0$$

Даний метод можна використовувати лише для цільової функції що диференціюється. Але і в цьому випадку можуть виникнути серйозні труднощі при вирішенні системи нелінійних рівнянь. У багатьох випадках ніякої формули для цільової функції немає, а є лише можливість визначення її значень в довільних точках розглянутої області за допомогою деякого обчислювального алгоритму або шляхом фізичних вимірювань. Завдання полягає в наближеному визначенні найменшого значення функції у всій області при відомих її значеннях в окремих точках.

Для вирішення практичних завдань пошуку мінімум цільової функції U, можна ввести дискретну кількість точок (вузлів) шляхом подрібнення інтервалів зміни параметрів $x_1, x_2, ..., x_n$ на частини з кроком $h_1, h_2, ..., h_n$. В отриманих вузлах можна визначити значення цільової функції і серед цих значень вибрати мінімальне. Даний метод може бути використаний для функції однієї змінної. В багатовимірних задачах оптимізації, де число проектних параметрів досягає п'яти і більше, цей метод вимагав би занадто великого обсягу обчислень.

Проведена оцінка показує, що подібні методи загального пошуку з використанням суцільного перебору для вирішення багатовимірних задач оптимізації не є ефективними. Необхідні спеціальні чисельні методи, засновані на цілеспрямованому пошуку. Розглянемо деякі з них.

Метод градієнтного спуску.

Нехай потрібно знайти найменше значення цільової функції $U = f(x_1, x_2, ..., x_n)$. В якості початкового наближення виберемо деяку точку M_0 в n-мірному просторі, с координатами $x_1^0, x_2^0, ..., x_n^0$. Зафіксуємо всі координати функції U, окрім першої. Тоді $U = f(x_1, x_2^0, ..., x_n^0)$ - є функція однієї змінної x_1.

Вирішуючи одновимірну задачу оптимізації для цієї функції, ми від точки M_0 переходимо до точки M_1 $(x_1^1, x_2^0, ..., x_n^0)$, в якій функція U приймає найменше значення за координатою x_1 при фіксованих інших координатах. У цьому полягає перший крок процесу оптимізації, що полягає в сходженні по координаті x_1.

Тепер зафіксуємо всі координати, окрім x_2, та розглянемо функцію цієї змінної $U = f(x_1^1, x_2, x_3^0, ..., x_n^0)$. Знову вирішуючи одновимірну задачу оптимізації, знаходимо її найменше значення при $x_2 = x_2^1$ тобто в точці M_2 $(x_1^1, x_2^1, x_3^0, ..., x_n^0)$. Аналогічно

проводитися спуск по координатам x_3, x_4, ..., x_n, потім процедура знову і знову повторюється від x_1 до x_n.

У результаті цього процесу виходить послідовність точок M_0, M_1, ..., в яких значення цільової функції складають монотонно спадну послідовність $f(M_0) \geq f(M_1) \geq f(M_2) \geq$ **.....** .

На любому k-ому кроці цей процес можна перервати, і значення $f(M_k)$ приймається в якості найменшого значення цільової функції в даній області.

Таким чином, метод градієнтного спуску зводить задачу про знаходження найменшого значення функції багатьох змінних до багаторазового вирішення одновимірних задач оптимізації по кожному проектному параметру. Даний метод легко проілюструвати геометрично для випадку функції двох змінних $z = f(x, y)$, що описує деяку поверхню в тривимірному просторі. На малюнку позначені лінії рівня цієї поверхні (рис. 1).

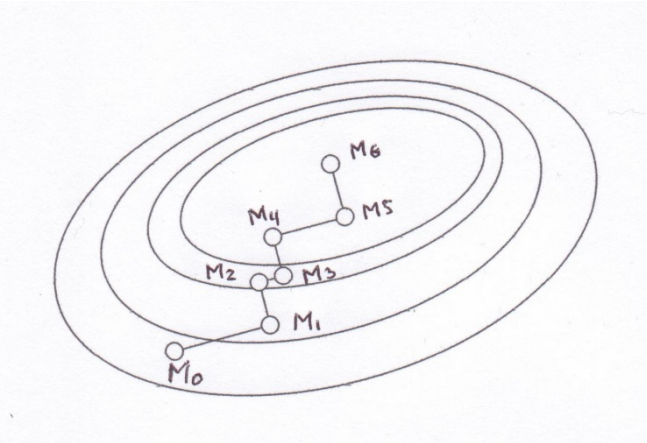

Рис. 1. Схематичне зображення градієнтного спуску

Процес оптимізації в цьому випадку проходить наступним чином. Точка $M_0(x_0, y_0)$ описує початкове наближення. Здійснюючи спуск по координаті x, потрапляємо в точку $M_1(x_1, y_0)$. Далі, рухаючись паралельно осі ординат, прийдемо в точку $M_2(x_1, y_1)$, рухаючись в напрямку градієнтного спуску. Важливим тут є питання про збіжність розглянутого процесу оптимізації. Іншими словами, чи буде послідовність значень цільової функції $f(M_0)$, $f(M_1)$, ... сходитися до найменшого її значенням в даній області? Це залежить від виду самої функції і вибору початкового наближення.

Метод найшвидшого спуску із дробленням кроку.

У цьому варіанті градієнтного методу величина кроку на кожній ітерації обирається за умови, що будь-який, черговий крок наближає нас до екстремуму гіперболоїда, а значить і до відповідного значенням координат потрібної точки. Ця процедура відбувається наступним чином. Обирається ймовірне значення координат потрібної точки і деякий початковий крок. Тепер для кожного k-кроку вважають, що його напрямок збігається з напрямком в бік точки максимуму функції і роблять крок градієнтного методу. Якщо умова виконується, то переходять до наступного k. Якщо ж не виконується, то множать значення кроку на коефіцієнт корекції («дроблять крок») і повторюють цю процедуру до тих пір, поки нерівність не буде виконуватися. Ця процедура для кожного k за кінцеве число кроків приводить нас до потрібного результату. Описаний алгоритм позбавляє нас від проблеми вибору значення кроку на кожному k, замінюючи її на проблему вибору ймовірних координат шуканої точки і величини стартового кроку, до яких градієнтний метод менш чутливий. При цьому, зрозуміло, що обсяг обчислень зростає (у зв'язку з необхідністю процедури дроблення кроку), втім, не дуже сильно, оскільки в більшості завдань основні обчислювальні витрати лягають на обчислення градієнта (вектора).

Формулювання цілей статті. Вивчити можливість створення системи визначення координат джерела звуку (пострілу) на базі мобільного пристрою у взаємодії з БПЛА. Визначити і реалізувати алгоритм побудови мобільного додатку під Android що обчислює координати пострілу. За допомогою розробленого прототипу програми перевірити працездатність даного алгоритму на смартфоні. Дослідити вплив схеми розстановки мікрофонів на БПЛА на швидкість визначення координат пострілу і шляхи підвищення точність вихідних даних.

Виклад основного матеріалу. За аналогією з принципом побудови супутникової системи навігації – GPS (англ. *Global Positioning System* – система глобального позиціонування) (рис. 2), побудуємо графічну схему розташування об'єктів моделі обчислення координат джерела пострілу (рис. 3).

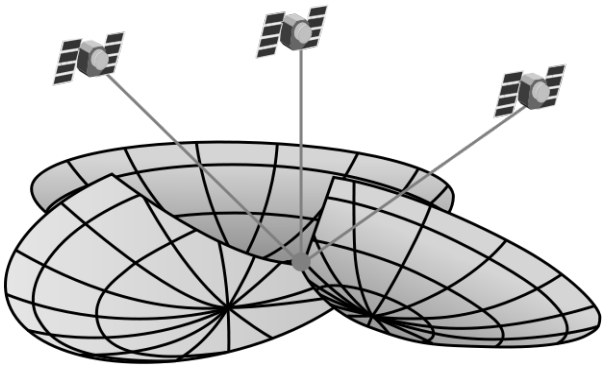

Рис. 2. Схема побудови супутникової системи навігації – GPS
Де: A, B, C, D, …. – мікрофони зі своїми координатами;
S – джерело звука (ДЗ);

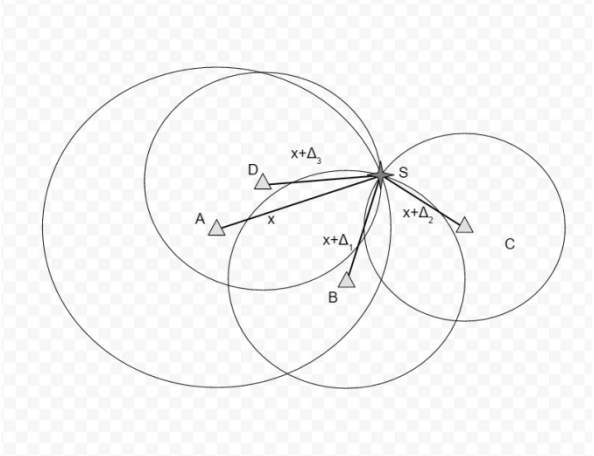

Рис. 3. Схема розташування об'єктів моделі обчислення координат джерела пострілу

X = відстань від S до A

Δ_1= швидкість звуку * різниця в часі отримання сигналу між A та B

Δ_2= швидкість звуку * різниця в часі отримання сигналу між A та C

Δ_3= швидкість звуку * різниця в часі отримання сигналу між A та D

І так само, по аналогії з GPS, визначення координат пострілу пропонується проводити шляхом вимірювання моментів часу прийому сигналу мікрофонами від джерела звуку S. Для визначення тривимірних координат джерела пострілу нам необхідно мати чотири рівняння даних. При цьому відстань дорівнює добутку швидкості звуку на різницю моментів прийому сигналу мікрофонами та моменту його синхронного випромінювання від джерела звуку.

Побудуємо чотирирівневу систему нелінійних рівнянь.

$\| S - A \| = x$

$\| S - B \| = x + \Delta_1$

$\| S - C \| = x + \Delta_2$

$\| S - D \| = x + \Delta_3$

$(S_x - A_x)^2 + (S_y - A_y)^2 + (S_z - A_z)^2 = x^2$

$(S_x - B_x)^2 + (S_y - B_y)^2 + (S_z - B_z)^2 = (x + \Delta_1)^2$

$(S_x - C_x)^2 + (S_y - C_y)^2 + (S_z - C_z)^2 = (x + \Delta_2)^2$

$(S_x - D_x)^2 + (S_y - D_y)^2 + (S_z - D_z)^2 = (x + \Delta_3)^2$

Ця система рівнянь, реалізована в якості алгоритму майбутньої програми, дає нам можливість визначити координати джерела звуку. Але вона є не практичною з точки зору її використання в умовах бойових дій, так як, працює без урахування шумів, перешкод і низького рівня корисного сигналу, а завдання визначення координат об'єкту може стати нестійким, і тому даний підхід несе на собі суто теоретичне навантаження.

У свою чергу в якості алгоритму для побудови мобільного додатку під Android обраний метод градієнтного спуску з дробленням кроку. Основна ідея методу градієнтного спуску полягає в тому, щоб здійснювати оптимізацію в напрямку найшвидшого спуску, а цей напрямок задається антиградієнтом.

На першому етапі роботи програми, в якості вхідних даних отримуємо масив з координатами мікрофонів, а також масив фіксації пострілу за часом на мікрофонах. Якщо координати джерела звуку невідомі, їх треба тим чи іншим способом підібрати, критерієм правильності підбору координат є мінімізація функціоналу Ф.

$$\Phi(x,y,z) = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \left(\sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} - \sqrt{(x_j - x)^2 + (y_j - y)^2 + (z_j - z)^2} - V_{зв} * t_{ij} \right)^2$$

З огляду на те, що нам не відомі реальні координати пострілу, припустимо, що джерело звуку розташовано на початку координат. Тоді в результаті підстановки вихідних даних в Ф отримуємо результат, що

характеризує похибку у визначенні реальних координат джерела пострілу.

```
double measurementError(Coordinates possibleCoord, int microphonesNumber,
Vector<Coordinates> microphone, Vector<Double> microphoneTime) {
double measurementErrorTemp = 0;
for (int i = 0; i < microphonesNumber - 1; i++) {
 for (int j = i + 1; j < microphonesNumber; j++) {
double microphonesDependency = sqrt(sqr(microphone.get(i).x - possibleCoord.x) +
  sqr(microphone.get(i).y - possibleCoord.y) + sqr(microphone.get(i).z - possibleCoord.z)) -
sqrt(sqr(microphone.get(j).x - possibleCoord.x) + sqr(microphone.get(j).y - possibleCoord.y) +
sqr(microphone.get(j).z - possibleCoord.z)) - soundSped * (microphoneTime.get(i) - microphoneTime.get(j));
measurementErrorTemp += microphonesDependency * microphonesDependency;
}
}
return measurementErrorTemp;
```

Далі, ми обчислюємо Ф’ як похідну від Ф та отримуємо вектор, який дає нам напрямок градієнтного спуску і його крок. По мірі наближення до екстремуму гіперболоїда похибка зменшується і в кінцевому підсумку, ми приходимо до бажаного результату з тією або іншою точністю.

$$\Phi'(x,y,z) = 2\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}\left(\sqrt{(x_i-x)^2+(y_i-y)^2+(z_i-z)^2}-\sqrt{(x_j-x)^2+(y_j-y)^2+(z_j-z)^2}-V_{зв}*t_{ij}\right)*\left(\frac{x-x_i}{\sqrt{(x_i-x)^2+(y_i-y)^2+(z_i-z)^2}}-\frac{x-x_j}{\sqrt{(x_j-x)^2+(y_j-y)^2+(z_j-z)^2}}\right)$$

```
Coordinates measurementErrorNegativeGradient(Coordinates possibleCoord, int microphonesNumber,
Vector<Coordinates> microphone, Vector<Double> microphoneTime) {
Coordinates updateCoordinate = new Coordinates(0, 0, 0);
for (int i = 0; i < microphonesNumber - 1; i++) {
for (int j = i + 1; j < microphonesNumber; j++) {
double soundToMicrophoneDistance_i = sqrt(sqr(microphone.get(i).x - possibleCoord.x) +
sqr(microphone.get(i).y - possibleCoord.y) + sqr(microphone.get(i).z - possibleCoord.z));
 double soundToMicrophoneDistance = sqrt(sqr(microphone.get(j).x - possibleCoord.x) +
sqr(microphone.get(j).y - possibleCoord.y) + sqr(microphone.get(j).z - possibleCoord.z));
double measurementError = soundToMicrophoneDistance_i -
soundToMicrophoneDistance - soundSped * (microphoneTime.get(i) -
microphoneTime.get(j));
 if (Math.abs(soundToMicrophoneDistance_i) > 0.0001 &&
Math.abs(soundToMicrophoneDistance) > 0.0001) {
double measurementError_x = ((possibleCoord.x - microphone.get(i).x) /
soundToMicrophoneDistance_i) - ((possibleCoord.x - microphone.get(j).x) /
soundToMicrophoneDistance);
double measurementError_y = ((possibleCoord.y - microphone.get(i).y) /
 soundToMicrophoneDistance_i) - ((possibleCoord.y - microphone.get(j).y) /
soundToMicrophoneDistance);
 double measurementError_z = ((possibleCoord.z - microphone.get(i).z) /
 soundToMicrophoneDistance_i) - ((possibleCoord.z - microphone.get(j).z) /
soundToMicrophoneDistance);
 updateCoordinate.x += measurementError * measurementError_x;
updateCoordinate.y += measurementError * measurementError_y;
updateCoordinate.z += measurementError * measurementError_z;
} else {
updateCoordinate.x += 1;
updateCoordinate.y += 1;
updateCoordinate.z += 1;
}
}
}
updateCoordinate.x = -updateCoordinate.x;
updateCoordinate.y = -updateCoordinate.y;
updateCoordinate.z = -updateCoordinate.z;
return updateCoordinate; // vector of gradient
}
```

Необхідна нам точність обчисленні координат пострілу досягається завдяки встановленому нами, порогу виходу з циклу, який, в свою чергу, відповідав би необхідної точності розрахунків.

```
public class Main {
public static void main(String[] args) {
Coordinates possibleCoordinate = new Coordinates(0, 100, 0);
Vector<Coordinates> microphone = new Vector<Coordinates>();
microphone.addElement(new Coordinates(0, 0, 0));
// Situation 5.5 +++ for a single UAV
microphone.addElement(new Coordinates(0, 0.25, 0));
// ∅ - 0.5м., sound source (0.4, 34)
microphone.addElement(new Coordinates(0, 0.5, 0));
// 5 microphones - 15 sec.
microphone.addElement(new Coordinates(0.25, 0.5, 0));
// 66 м. during 25 sec. / from 100м. to 34м.
microphone.addElement(new Coordinates(0.5, 0.5, 0));
int microphonesNumber = microphone.size();
Vector<Double> microphoneTime = new Vector<Double>();
microphoneTime.addElement(0.09992169284952675); // Situation 5.5
microphoneTime.addElement(0.09918707657332533);
microphoneTime.addElement(0.09845246106142029);
```

```
microphoneTime.addElement(0.0984464304540607);
microphoneTime.addElement(0.0984458821987095);
GradientDescent gradientD = new GradientDescent();
double measurementErrorValue = gradi-
entD.measurementError(possibleCoordinate,
microphonesNumber, microphone, microphoneTime);
double coefficient = 999999999;
double drobCoefficient = 0.99;
double precision = 0.0000000001; // 0.00000001 for ∅ 1м., 0.0000000001 for ∅ 0.5 м.
//boolean flag = true;
Coordinates negativeGradientValue;
    while (measurementErrorValue > precision) {
negativeGradientValue =
gradi-
entD.measurementErrorNegativeGradient(possibleCoordi
nate,
microphonesNumber, microphone, microphoneTime);
Coordinates newPossibleCoordinate = new Coordinates(
possibleCoordinate.x + coefficient * negativeGradi-
entValue.x,
possibleCoordinate.y + coefficient * negativeGradi-
entValue.y,
possibleCoordinate.z + coefficient * negativeGradi-
entValue.z);
    double newMeasurementErrorValue =
gradientD.measurementError(newPossibleCoordinate,
microphonesNumber, microphone, microphoneTime);
if (newMeasurementErrorValue < measurementErrorVal-
ue) {
possibleCoordinate = newPossibleCoordinate;
measurementErrorValue = newMeasurementErrorValue;
} else {
coefficient *= drobCoefficient;
}
System.out.println("пр: " + possibleCoordinate +
", ош: " + measurementErrorValue + ", кф: " + coeffi-
cient);
}
System.out.println("источник: " + possibleCoordinate);
}
}
```

Виділення не вирішених раніше частин загальної проблеми.

Серед раніше невирішених проблем слід відзначити проблему, пов'язану з нестабільними показниками часу обчислення кінцевих результатів. Причиною такої нестабільності є наявність двох екстремумів в графіку досліджуваної функції (рис. 4, а). Це явище проявляється в ситуації коли відстань між мікрофонами значно менше ніж відстань від БПЛА до джерела звуку. За для вирішення даної проблеми необхідно, задаючи ймовірні координати пострілу, обирати позицію на проміжку між БПЛА і джерелом звуку. Та навпаки, якщо згадані відстані не суттєво відрізняються, ми спостерігаємо тільки один екстремум функції (рис. 4, б). З метою збільшення швидкості обчислень ймовірні координати пострілу повинні бути максимально наближеними до реально розташованих вогневих точок противника. Так, наприклад прицільний вогонь з автомата ведеться з відстані 500 м, з кулемети 600–800 м. а снайперської гвинтівки 800–1000 м. З цією метою в розробленому додатку передбачена можливість вибору типу зброї, для якої обчислюються координати. Вірний вибір даних параметрів, безумовно, залежить від досвіду командира підрозділу.

Я

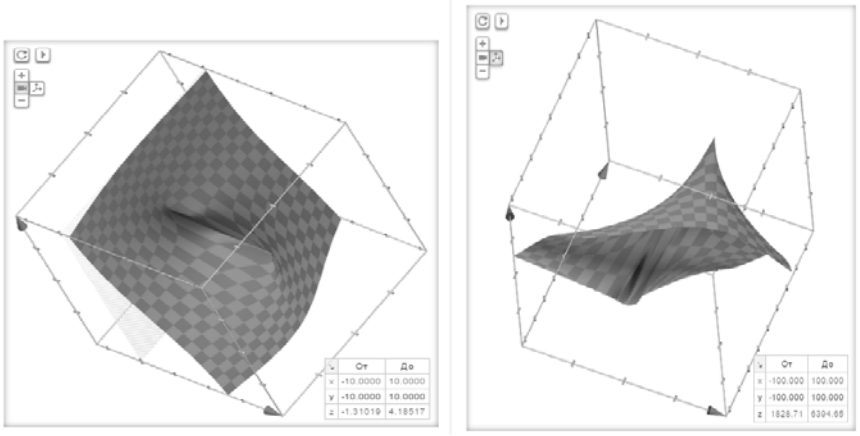

Рис. 4. Графіки функції: а – з двома екстремумами; б – з одним екстремумом

Висновки. Аналіз викладених підходів та результати їх практичного втілення підтверджують можливість створення системи визначення координат джерела звуку (пострілу) на базі мобільного пристрою у взаємодії з БПЛА. Розроблений мобільний додаток під Android SoundRadarCopterApp з реалізацією метода градієнтного спуску в якості алгоритму обчислює координати пострілу підтвердив свою ефективність під час тестування на смартфоні (рис. 5). Теоретичні та експериментальні дослідження довели, що схема розстановки мікрофонів на БПЛА, їх кількість та відстань між ними має безпосередній вплив на швидкість визначення координат пострілу і шляхи підвищення точність вихідних даних.

З'ясовано, що мікрофони в кількості не менше 5 шт. (5–8), слід рівномірно розташовувати в горизонтальній площині вздовж кола діаметром від 0,5 до 1 метра на базі одного БПЛА.

Це дозволяє з швидкістю від декількох секунд до декількох десятків секунд, в залежності від обраних, вірогідних координат пострілу, підрахувати реальні координати джерела звуку с точністю до декількох сантиметрів. Так, при очкуванні (4, 35, 0) маємо наступні значення:

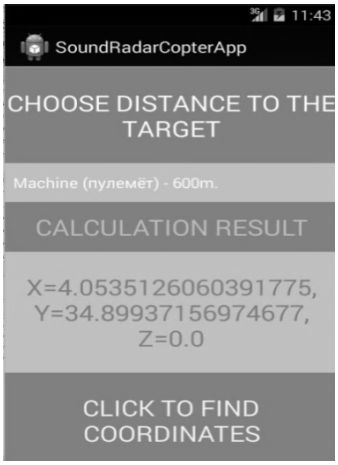

Рис. 5. Знімки екрана смартфона (скріншоти)

ЛІТЕРАТУРА

1. Бойовий статут сухопутних військ частина 3 (взвод, відділення, танк) – командування сухопутних військ збройних сил України. – Київ. – 2010
2. А. В. Львов, М. Н. Агапов, А. И. Тищенко Триангуляционная система определения координат источника звука – Алтайский гостехуниверситет, Ползуновский вестник №22010.
3. Градиентные методы [Електронний ресурс]. – Режим доступу : Url: http://www.sbras.ru/rus/textbooks/akhmerov/mo_unicode/3.html (дата звернення: 10.04.2016).
4. Метод градиентного спуска [Електронний ресурс] Url: http://www.machinelearning.ru/wiki/index.php?Title= метод градиентного спуска (дата звернення: 18.04.2016).
5. Gps spherical location [Електронний ресурс] Url: https://commons.wikimedia.org/wiki/file:gps_spheres.svg (дата звернення: 20.03.2016).
6. Ж. О. Бєлозьоров Методи виявлення та відображення на картах рухомих об’єктів за даними мобільної гетерогенної мережі в системах ведення бою – Чорноморський національний університет імені Петра Погили, могилянські читання – 2015, збірник тез – том 1.
7. Ж. О. Бєлозьоров Взаємодія командира механізованого відділення та бпла – Чорноморський національний університет імені Петра Могили, Ольвійський форум – 2016, збірник тез-том5.
8. М. П. Мусиенко, И. Н. Журавская, И. С. Бурлаченко, А. О. Денисов, А. О. Корецкая, Ж. О. Белозёров, Мобильные мониторинговые сети критического применения: проблемы создания и направления развития. – Черноморский Государственный университет имени Петра Могилы, серія : комп’ютерні технології. – 2015. – т. 266, – Режим доступу : http://nbuv.gov.ua/ujrn/npchduct_2015_266_254_19

Ж. О. Белозеров,
Черноморский национальный университет
им. Петра Могилы,
г. Николаев, Украина

АНАЛИЗ АЛГОРИТМА ВЫЧИСЛЕНИЯ КООРДИНАТ ВЫСТРЕЛА И ЕГО ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ НА БАЗЕ МОБИЛЬНОГО УСТРОЙСТВА ВО ВЗАИМОДЕЙСТВИИ С БПЛА

В статье изучается возможность создания системы определения координат источника звука (выстрела) на базе мобильного устройства во взаимодействии с БПЛА. Предложен и реализован метода градиентного спуска с дроблением шага в качестве алгоритма для построения мобильного приложения под Android. С помощью разработанного прототипа программы проверена работоспособность данного алгоритма на смартфоне. Экспериментальным путем исследовано влия-

ние схемы расстановки микрофонов на БПЛА на скорость определения координат выстрела и пути повышения точность выходных данных.

Ключевые слова: *БПЛА; координата; источник звука; микрофон; смартфон.*

Zh. O. Byelozyorov,
Petro Mohyla Black Sea National University,
Mykolayiv, Ukraine

ANALYSIS OF THE ALGORITHM FOR CALCULATING THE COORDINATES OF THE SHOT AND ITS PRACTICAL IMPLEMENTATION ON THE BASIS OF MOBILE DEVICE IN COOPERATION WITH THE UAV

We study the possibility of creating a sound source positioning system (shot) on the basis of the mobile device in cooperation with UAV. The method of gradient descent with a crushing step is proposed and implemented in an algorithm for building mobile applications for Android. With the help of the developed prototype program has confirmed the operation of the algorithm by a Smartphone. The effect of the scheme of arrangement of microphones on the UAV was experimentally studied to determine the speed of on the shot coordinates and ways to improve the accuracy of the output data.

Key words: *drones; coordinate; sound source; microphone; Smartphone.*

Рецензенти: д. п. н., проф. ***О. П. Мещанінов;***
д. т. н., проф. ***А. Н. Хомченко.***

© Бєлозьоров Ж. О., 2016

Дата надходження статті до редколегії 02.09.16