Bulletin of the *Transilvania* University of Braşov • Vol. 17(66) No. 1 – 2024
Series I: Engineering Sciences
https://doi.org/10.31926/but.ens.2024.17.66.1.1

DETERMINATION OF CENTERS OF WEIGHT THROUGH THE ANALYTICAL METHOD AND VALIDATION OF THE RESULTS THROUGH THE C# PROGRAMMING LANGUAGE

R. M. BLEOTU[1] C. PREDA[2]

Abstract: *This paper focuses on the development and implementation of an analytical method for determining the centers of gravity of complex bodies. Through this process, the aim is to obtain an accurate and efficient solution for determining the center of gravity in a practical and easy-to-achieve way. The paper includes a detailed presentation of the analytical method used, including the algorithms and mathematical formulas involved in determining the center of gravity for various types of complex bodies. It also examines how this method can be implemented in the C# programming language to allow validation of the results obtained from the analytical calculation.*

Key words: *centers of gravity, application, programming, analytical calculus, complex bodies.*

1. Introduction

One of the most important aspects in assessing the condition of a body or systems of bodies is determining the center of gravity. However, before discussing about centers of gravity, it is necessary to mention the centers of mass. Centers of mass are intrinsic features of material point systems.

The center of gravity of a body is a crucial parameter in many fields of science and engineering, especially in disciplines such as aerospace, automotive design, robotics, civil engineering, and biomechanics. It represents the point at which the entire weight of an object can be considered to act, and it is essential for ensuring balance, stability, and performance in both static and dynamic systems [12], [14].

The position of the center of mass depends on the way the mass is distributed in the system of material points and is located in the area with the greater mass [4]. In the

[1] Department of Industrial Machinery and Equipment, Faculty of Engineering, *Lucian Blaga* University of Sibiu, Romania, robert.bleotu@ulbsibiu.ro
[2] Department of Industrial Machinery and Equipment, Faculty of Engineering, *Lucian Blaga* University of Sibiu, Romania

calculation of the centers of mass, several generally valid properties are used in the calculation of the centers of gravity [14]:

- if the sizes of all masses are multiplied or reduced by the same scalar $\lambda \neq 0$, the position of the center of mass does not change;
- if a material system has a plane of symmetry, then the center of mass is in that plane [14, 15];
- if a material system has an axis of symmetry, then the center of mass is on that axis;
- if a material system has a pole of symmetry, then the center of mass is at that pole.

Despite all these similarities, there is an important difference between centers of mass and centers of gravity. The center of mass is always located somewhere inside the body, which is not generally true in the case of centers of gravity, for the latter there is the possibility of being placed somewhere outside the body [3]. The center of gravity of a body of measurable dimensions is the point at which its weight is considered to be applied. It is therefore one of the fundamental concepts of statics [15]. Therefore, the center of gravity is the geometric place where the entire mass of a body or systems of bodies is concentrated, its determination being crucial for the analysis and design of structures, vehicles and other physical systems, as it helps to calculate balance, stability and maneuverability. Therefore, it can be shown, both theoretically and experimentally, that the position of the center of gravity is well determined for each body and does not depend on its orientation in space [9].

In applications like drones or aircraft, the center of gravity influences flight stability and control. A well-positioned center ensures that an aircraft can maintain its desired orientation and perform efficiently in various flight conditions. In industrial robots or autonomous systems, it plays a role in improving the precision of movements and reducing mechanical stress. As such, the ability to determine the center of gravity is not just a theoretical factor, but also a practical necessity that directly affects the design, operation, and safety of modern technologies.

The problem addressed in this paper is the determination of centers of gravity for complex bodies, a topic of fundamental importance in fields such as mechanical engineering, robotics, and industrial design [14]. In this problem, ways to determine the exact position of the center of gravity for various types of complex bodies, which can have varying shapes and sizes, are investigated. The proposed method must provide accurate results and allow their validation using the C# programming language.

2. The purpose of the work

The primary goal of this paper is to develop and implement an analytical method for determining the center of gravity of complex bodies and validate the results through a custom-built software application written in the C# programming language. This work seeks to address the challenge of accurately calculating the center in practical applications where mass distribution can be irregular or affected by changing components [2], [4].

By integrating theoretical and practical approaches, this paper aims to approve the analytical calculations with practical, technology-driven solutions for determining the center of gravity. The methodology is designed to be flexible and applicable to a wide

range of complex bodies, allowing for its use in diverse fields such as robotics, aerospace, and mechanical design. Lately, this study demonstrates the benefits of using modern programming techniques to enhance the accuracy, efficiency, and practicality of center of gravity determination.

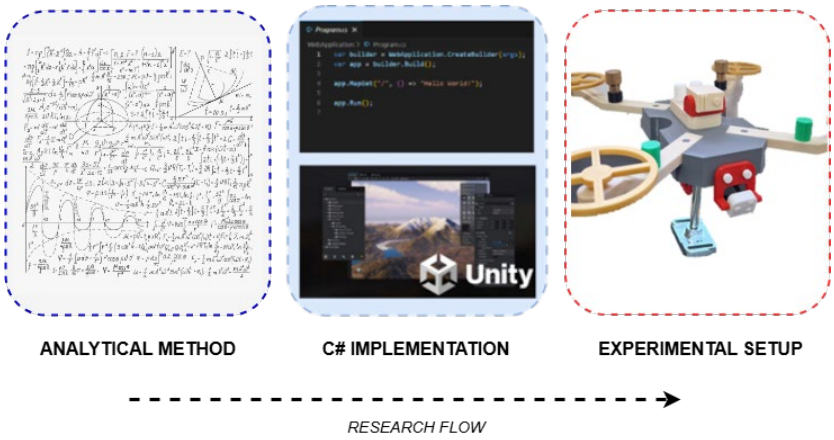

Fig.1. *Objectives for the paper diagram*

Specific objectives of the scientific paper, which are presented also in Figure 1:

Develop an analytical method: the first aim is to explore and formulate a detailed analytical method based on mathematical principles for calculating the center. This will involve deriving equations for various types of simple and complex bodies using principles of integration and summation.

Apply mathematical principles to real-world objects: another key goal is to apply these mathematical formulas to practical situations where the center of gravity needs to be accurately determined, including objects with varying shapes, mass distributions, and configurations. For instance, this paper will examine a drone model with adjustable weights, modeling how it shifts with these changes.

Build a C# application for center of gravity determination: another objective is to develop a software tool that automates the process of calculating it using C#. This application will provide for person with a fast and accurate tool for center of gravity calculation, capable of handling both simple and complex bodies. The program will integrate the theoretical principles into a user-friendly format, allowing real-time center computation for different configurations.

Validate results with numerical and software methods: the final objective of the study is to validate the accuracy of the analytical method by comparing results with those generated through the C# program. This ensures that both theoretical approaches and software applications yield consistent and reliable results. The validation process will also involve physical testing, such as constructing a drone model with variable shapes, to confirm the theoretical center of gravity calculations align with real-world observations [10].

Lately, this work aims to provide a comprehensive, accurate, and practical solution for determining the center of gravity of complex systems. The paper offers another

combination of theoretical analysis and practical implementation, contributing to fields in engineering, where the precise knowledge of it is crucial for stability, performance, and efficiency.

3. Theoretical framework

3.1 Mathematical principles and equations for center of gravity calculation

The center of gravity is defined as the point at which the entire weight of a body or system is concentrated [12]. Mathematically, the center of gravity is a weighted average of the distribution of mass in an object. For simple bodies, the center of gravity can often be determined through geometric properties [1, 5]. However, for complex bodies with irregular shapes and varying mass distributions, the calculation becomes more complicated and requires the use of integrals, summations, and advanced geometrical analysis.

For a rigid body, the center of gravity is determined by considering the distribution of mass within the object [1]. This involves calculating the weighted average of the positions of the various mass elements in the body. Mathematically, the center is the point where the object's moments about each axis ($x-axis, y-axis, z-axis$) are balanced.

Center of gravity in at two-dimensional plane

For a two-dimensional object with mass distributed across its area, the coordinates of the center of gravity (x_{cg}, y_{cg}) are given by the following equations:

$$x_{cg} = \frac{\int x dm}{\int dm}; y_{cg} = \frac{\int y dm}{\int dm} \tag{1}$$

where, x, y are the coordinates of differential mass elements dm and $\int dm$ is the total mass of the object.

In cases where the object has uniform density, it can be simplified $\int dm = \sigma dA$, σ being the surface mass density and dA is a differential area element. For a constant σ, the center of gravity simplifies to:

$$x_{cg} = \frac{\int x dA}{A};\ y_{cg} = \frac{\int y dA}{A} \tag{2}$$

where A is the total area of the object. These integrals can be solved for specific geometric shapes using standard integration techniques.

Center of gravity in three-dimensional space

For three-dimensional objects, the center of gravity coordinates (x_{cg}, y_{cg}, z_{cg}) are determined by integrating over the volume of the object. The equations for the CoG in 3D space are:

$$x_{cg} = \frac{\int x dm}{m}; y_{cg} = \frac{\int y dm}{m}; z_{cg} = \frac{\int z dm}{m} \tag{3}$$

Again, for uniform density ρ, where $dm = \rho \mathrm{d}V$ and $\mathrm{d}V$ is the differential volume element, these equations simplify to [8]:

$$x_{cg} = \frac{\int x dV}{V};\ y_{cg} = \frac{\int y dV}{V};\ z_{cg} = \frac{\int z dV}{V} \tag{4}$$

where V is the total volume of the object.

Center of gravity for composite bodies

Many real-world objects are made of multiple distinct parts, each with its own mass and center of gravity. For composite systems, the overall center of gravity is determined by finding the weighted average of the centers of gravity of the individual parts [6]. Suppose an object consists of *n* parts, each with mass m_i and center of gravity (x_i, y_i, z_i). The overall center of gravity is given by the following summation:

$$x_{cg} = \frac{\sum_{i=1}^{n} m_i x_i}{\sum_{i=1}^{n} m_i};\ y_{cg} = \frac{\sum_{i=1}^{n} m_i y_i}{\sum_{i=1}^{n} m_i};\ z_{cg} = \frac{\sum_{i=1}^{n} m_i z_i}{\sum_{i=1}^{n} m_i} \tag{5}$$

This method is especially useful for calculating the center of gravity of complex machines, structures, or multi-body systems, where each component has a well-defined mass and center.

Moment of inertia and center of gravity

The calculation of the center of gravity is closely related to the concept of **moment of inertia**. The moment of inertia quantifies the resistance of a body to rotational motion about an axis. The moment of inertia III with respect to a given axis is calculated as:

$$I = \int_V r^2 \rho(x, y, z) dV \tag{6}$$

where r is the perpendicular distance from the axis to the mass element $dm = \rho(x, y, z)dV$.

The moment of inertia is an important consideration in determining the stability of objects in motion and is directly related to the distribution of mass around the center of gravity [12].

3.2. Integrals and summations for simple and complex bodies

The determination of the center of gravity requires calculating how mass is distributed throughout an object. For simple bodies with regular shapes, summations may change, while for complex or continuous mass distributions, integrals are necessary. This chapter delves into both integrals and summations, providing theoretical background, explanations, and detailed formulas to calculate the center of gravity for various objects.

When working with discrete systems or simple bodies the center of gravity can often be calculated using summations rather than integrals. This method is especially useful when

dealing with objects made up of multiple distinct parts or when the mass distribution is concentrated at known points.

For a system of n point masses $m1, m2, \dots, mn$, the coordinates of the center of gravity x_{cg}, y_{cg}, z_{cg} are found by calculating the weighted average of the positions of each mass.

Integral approach for continuous bodies

For continuous bodies, where mass is distributed over a volume, surface, or line, summation is replaced by integration. The integration approach divides the object into infinitesimally small mass elements and sums their contributions to the overall center of gravity [7]. The integrals consider both the geometry and the mass distribution of the object.

For an object with mass distributed along a one-dimensional curve (*e.g., a rod*), the center of gravity is calculated as follows. Suppose the rod is aligned along the x-axis, with mass per unit length, and its length spans from x=a to x = b. The center of gravity is given by:

$$x_{cg} = \frac{\int_a^b x\lambda(x)dx}{\int_a^b \lambda(x)dx} \tag{7}$$

where $\lambda(x)$ is the mass per unit length at position x, $\int_a^b \lambda(x)dx$ gives the total mass of the rod.

If the rod has a uniform mass distribution, meaning $\lambda(x) = \lambda(0)$, the integrals simplify:

$$x_{cg} = \frac{\lambda_0 \int_a^b xdx}{\lambda_0 \int_a^b dx} = \frac{\frac{1}{2}(b^2-a^2)}{b-a} = \frac{a+b}{2} \tag{8}$$

This result shows that for a uniformly distributed rod, the center of gravity is located at the midpoint.

4. Methodology

This chapter provides a detailed account of the procedures and methods used to achieve the objectives outlined in this paper, which were described in the previous chapter. The methodology is focused on combining mathematical analysis with modern digital tools to determine the center of gravity for complex bodies. The process involves an analytical approach to solving center problems, 3D modeling of real-world assemblies, and the implementation of a C# application for accurate and efficient calculations. The methodology ensures that both theoretical and practical aspects of its determination are covered comprehensively.

4.1. Analytical method

To determine the center of gravity of the assembly, five main steps were taken. In the first step, the main body was divided into simple geometric bodies, and their center of

gravity was calculated relative to a unique reference system. In Figure 2 it is visible the division of the assembly into complex geometric bodies.

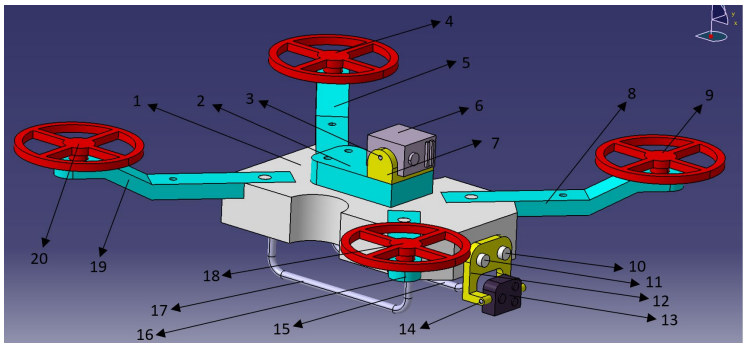

Fig.2. *Dividing the ensemble into complex geometric bodies*

The coordinates of the center of gravity of a simple body are represented by the notations x_i, y_i, z_i. For ease of calculation, we opted for a tabular representation of the data. Figure 3 illustrates the division of one of the complex geometric bodies into simple geometric bodies.

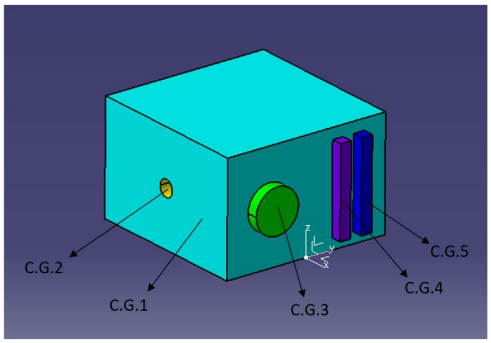

Fig.3. *Dividing the complex body into simple geometric bodies*

In the second step, the volume of each simple geometric body was calculated, followed by a multiplication operation with the coordinates of the center of gravity found [15]. This multiplication operation contributes to the final calculation. Table 1 shows a calculation model applied to body 6, the table below presents the coordinates, volumes, and products used in calculating the center of gravity for various bodies in the drone model.

Brief description of the data in the table for body 6 Table 1

Body	x_i	y_i	z_i	V_i	$V_i \times x_i$	$V_i \times y_i$	$V_i \times z_i$
C.G.1	-1.5	0	1	17.94	-26.91	0	17.94
C.G.2	-1.5	0	1	-0.2113	0.31702	0	-0.2113
C.G.3	0.1	-0.7	1	0.1005	0.01005	-0.0703	0.1005
C.G.4	0.1	0.595	1	0.064	0.0064	0.03808	0.064
C.G.5	0.1	0.995	1	0.064	0.0064	0.06368	0.064

The calculated values for each body in the drone model are used to determine their contributions to the overall center of gravity, and these products are used to determine the weighted contributions of each body to the overall center of gravity [5], [13].

In the third step, the volumes (V_i) calculated for each body geometry were summed, and then sums were performed on each axis. In the next step, the center of gravity for the complex body was calculated [2]. In the fourth step, the center of gravity for body 6 was determined, as can be seen in Table 2. Finally, in the last step, a unique reference system was chosen, relative to which the coordinates of the center of gravity were calculated weight of the analyzed complex body.

Coordinates for the body number 6 Table 2

Calculation formula	$Xc = \frac{\sum_{i=1}^{n} Xi * Vi}{\sum_{i=1}^{n} Vi}$	$Yc = \frac{\sum_{i=1}^{n} Yi * Vi}{\sum_{i=1}^{n} Vi}$	$Zc = \frac{\sum_{i=1}^{n} Zi * Vi}{\sum_{i=1}^{n} Vi}$
Result	-1.47964	0.001749	1

4.2. 3D modeling of a complex assembly

After performing the analytical calculation, the 3D model of the complex body was created. It was made with the help of the CATIA V5 program. Program that also allows the automatic determination of the center of gravity. The 3D model can be seen in Figure 4.

This chapter focuses on the 3D modeling of a complex assembly, designed specifically to represent a drone with various components. The objective is to create a realistic and detailed 3D model composed of multiple interconnected parts, allowing for accurate calculation and analysis of the center of gravity. This process is essential to simulate real-world conditions and provide input for both analytical and software-based center of gravity calculations.

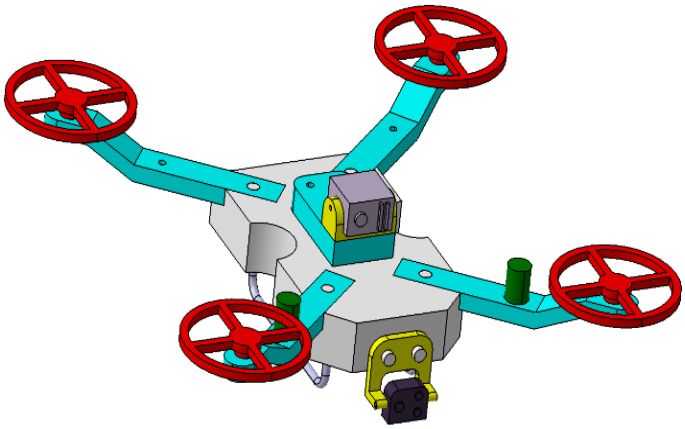

Fig.4. *Model 3D- complex body (drone)*

The complex assembly used for this study consists of a drone, which has been subdivided into 20 distinct bodies. These bodies represent different structural and functional components of the drone.

The rationale behind dividing the drone into 20 separate parts is to allow for the flexibility to analyze each component's contribution to the overall center. By modeling these components individually, it can be better understood how variations in mass distribution, weight placement, and geometry affect the entire assembly.

The 3D model not only served as a digital representation for center of gravity calculations, but also offered insights into practical design considerations for drones.

4.3. Realization of the application for determining the center of gravity

In this chapter, the development of an application designed to determine the center of gravity for complex assemblies, particularly drones, is discussed in detail. The application is built using the Unity platform, a widely-used game development environment known for its robust 3D modeling, real-time simulation capabilities, and support for advanced physics engines. Unity was chosen because it allows for both the visual representation of 3D models and the integration of programming languages, such as C#, to perform complex calculations. The role of the application is twofold: to serve as a practical tool for real-time center of gravity determination and to validate the theoretical calculations performed in earlier stages.

The primary goal of the application is to provide a practical interface for calculating and visualizing the center of a complex assembly, such as the 20-part drone described earlier. The secondary goal is to validate the theoretical calculations derived from the analytical method by providing a digital environment where users can interact with the model, adjust parameters, and observe changes to it in real-time. This application fills the gap between theory and practice, enabling designers and engineers to perform center calculations efficiently.

Algorithm 1 *for application interface designed*

```
using System.Collections;
using System.Collections.Generic;
using JetBrains.Annotations;
using UnityEngine;
public class centruGreutateDrona: MonoBehaviour
// Start is called before the first frame update Vector3 COM = Vector3.zero;
      public GameObject assembly; GameObject sageata; float cof;
         void Start()
      foreach(Transform part in
         assembly.GetComponentsInChildren<Transform>()){
}
// add rigidbody to each part
     if(part.GetComponent<Rigidbody>() == null){ print(part.gameObject.name);
}
part.gameObject.AddComponent<Rigidbody>(); part.GetComponent<Rigidbody>().useGravity
= false;
Rigidbody componentRb = part.GetComponent<Rigidbody>(); COM +=
componentRb.worldCenterOfMass componentRb.mass; c+=
part.GetComponent<Rigidbody>().mass;
COM /= c;
sageata = Instantiate(Resources.Load<GameObject>("SageataRosie"), COM,
Quaternion.Euler(90, 0, 0));
}
// Update is called once per frame
       void Update()
COM - Vector3.zero;
       float cof;
foreach(Transform part in assembly.GetComponentsInChildren<Transform>()){
}
Rigidbody componentRb-part.GetComponent<Rigidbody>(); COM +=
componentRb.worldCenterOfMass componentRb.mass; c+=
part.GetComponent<Rigidbody>().mass;
COM /= c;
sageata.transform.position= COM;
}
```

For the development of the application, the Unity platform was used, recognized as one of the most popular game development environments globally. In the process of implementing the application for determining the center of gravity, advanced algorithms were used to precisely calculate the position and relative weight of each component of the complex body. By means of these algorithms, it was possible to accurately determine the center of gravity of the complex body. A positive aspect of this application is the ability to provide a visual and interactive simulation of the drone, thanks to the use of the Unity platform. This feature facilitated the rapid visualization and testing of various component configurations in order to optimize the stability of the complex body. Algorithm 1 shows a code used to implement this application, in which a rigid body was adopted for each part and the code lines for updating according to each shape.

4.3.1. Implementation in C#

This chapter focuses on the technical aspects of implementing the application for determining the center of gravity using the C# programming language within the Unity platform. The C# code handles all the main functionality of the application, including center calculations, real-time adjustments of component masses and interaction with the user interface. The implementation was designed to be both efficient and intuitive, allowing users to interact with the drone model dynamically while providing accurate its calculations.

Algorithm 2 *for determination of center of gravity*

```
using System.Collections;
using System.Collections.Generic;
using JetBrains.Annotations;
using UnityEngine;
using TMPro;
public class centruGreutateDrona : MonoBehaviour
{
   // Start is called before the first frame update
   Vector3 CoM = Vector3.zero;
   public GameObject assembly;
   public GameObject greutateAlbastra;
   public GameObject greutateVerde;
   public TMP_Text pozitieCoMlabel;
   public TMP_Text greutateAlbastraCoMlabel;
   public TMP_Text greutateVerdeCoMlabel;
   GameObject sageata;
   float c = 0f;
   void Start()
   {
      foreach(Transform part in assembly.GetComponentsInChildren<Transform>()){
               // add rigidbody to each part
         if(part.GetComponent<Rigidbody>() == null){
            part.gameObject.AddComponent<Rigidbody>();
            part.GetComponent<Rigidbody>().useGravity = false;
            if(part.gameObject.name == "brat"){
               part.GetComponent<Rigidbody>().mass = 0.013f;
            }
            else if (part.gameObject.name == "camera"){
               part.GetComponent<Rigidbody>().mass = 0.006f;
            }
            else if (part.gameObject.name == "picior"){
               part.gameObject.GetComponent<Rigidbody>().mass = 0.003f;
            }
            else if(part.gameObject.name == "corp"){
               part.gameObject.GetComponent<Rigidbody>().mass = 0.014f;
            }
            else if(part.gameObject.name == "cam2"){
```

```
            part.GetComponent<Rigidbody>().mass = 0.006f;
          }
          else if (part.gameObject.name == "suportcam2"){
            part.GetComponent<Rigidbody>().mass = 0.006f;
          }
          else if (part.gameObject.name == "suportcam1"){
            part.GetComponent<Rigidbody>().mass = 0.004f;
          }
          else if (part.gameObject.name == "pin"){
            part.gameObject.GetComponent<Rigidbody>().mass = 0.001f;
          }
          else if (part.gameObject.name == "Component6:1" || part.gameObject.name ==
 "Component6:2"){
            part.gameObject.GetComponent<Rigidbody>().mass = 0.004f;
          }
          else {
            part.gameObject.GetComponent<Rigidbody>().mass = 0.0001f;
          }
            }
        Rigidbody componentRb = part.GetComponent<Rigidbody>();
        CoM += componentRb.worldCenterOfMass * componentRb.mass;
        c += part.GetComponent<Rigidbody>().mass;
      }
      CoM /= c;
      sageata = Instantiate(Resources.Load<GameObject>("SageataRosie"), CoM,
 Quaternion.Euler(90, 0, 0));
    }
    // Update is called once per frame
    void Update()
    {
      CoM = Vector3.zero;
      float c = 0f;
}
```

The algorithm presented above provides insights into how each part of the code works to compute the CoG, along with a real-time visual representation. This approach is particularly useful for assessing the balance and stability of the drone model, as the CoG is a central factor in its flight dynamics and performance. The algorithm supports dynamic adjustment, meaning that changes in the mass or configuration of drone components can be instantly reflected in the CoG. This is particularly useful for examining how real-world scenarios—such as adding payload or adjusting parts—would impact drone balance, which is a main goal for the upcoming research for the current study.

In the Algorithm 3 below, the primary objective of this algorithm is to allow dynamic adjustments to the masses of specified components—particularly the blue and green weights—within the 3D drone model. This enables real-time recalculations of the drone's center of gravity based on user input for mass values and desired CoG coordinates.

Algorithm 3 *for change in mass of bodies*

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;
using System;
public class SchimbaGreutate : MonoBehaviour
{
   public TMP_InputField inputGreutateAlbastra;
   public TMP_InputField inputGreutateVerde;
   public TMP_InputField inputCoordonateX;
   public TMP_InputField inputCoordonateY;
   public TMP_InputField inputCoordonateZ;
   public GameObject greutateAlbastra;
   public GameObject greutateVerde;
   public GameObject assembly;
   public void Aplica()
   {
      try
      {
         greutateAlbastra.GetComponent<Rigidbody>().mass =
Convert.ToSingle(inputGreutateAlbastra.text);
      }
      catch
      {
         greutateAlbastra.GetComponent<Rigidbody>().mass = 1;
      }
      try
      {
         greutateVerde.GetComponent<Rigidbody>().mass =
Convert.ToSingle(inputGreutateVerde.text);
      }
      catch
      {
         greutateVerde.GetComponent<Rigidbody>().mass = 1;
      }
   }
   public void SetareCentruDeGreutate()
   {
      Vector3 CoMDorit = new Vector3();
      try
      {
         CoMDorit.x = Convert.ToSingle(inputCoordonateX.text);
         CoMDorit.y = Convert.ToSingle(inputCoordonateY.text);
         CoMDorit.z = Convert.ToSingle(inputCoordonateZ.text);
      }
      catch
      {
```

```
        Debug.LogError("Coordonatele introduse nu sunt valide");
        return;
      }
      CalculeazaMasePentruCoM(CoMDorit);
    }
    void Start()
    {
      greutateAlbastra.GetComponent<Rigidbody>().mass = 0.1f;
      greutateVerde.GetComponent<Rigidbody>().mass = 0.1f;
      inputGreutateAlbastra.interactable = true;
      inputGreutateVerde.interactable = true;
      inputGreutateAlbastra.text = "0.1";
      inputGreutateVerde.text = "0.1";
    }
    void CalculeazaMasePentruCoM(Vector3 CoMDorit)
    {
      float totalMass = 0f;
      Vector3 currentCoM = Vector3.zero;
          foreach (Transform part in assembly.GetComponentsInChildren<Transform>())
      {
        Rigidbody componentRb = part.GetComponent<Rigidbody>();
        totalMass += componentRb.mass;
        currentCoM += componentRb.worldCenterOfMass * componentRb.mass;
      }
      currentCoM /= totalMass;
      Vector3 offset = CoMDorit - currentCoM;
      Vector3 positionAlbastra = greutateAlbastra.transform.position;
      Vector3 positionVerde = greutateVerde.transform.position;
      float distanceAlbastra = Vector3.Distance(positionAlbastra, CoMDorit);
      float distanceVerde = Vector3.Distance(positionVerde, CoMDorit);
      float masaNouaAlbastra = greutateAlbastra.GetComponent<Rigidbody>().mass +
 offset.magnitude * distanceVerde / (distanceAlbastra + distanceVerde);
      float masaNouaVerde = greutateVerde.GetComponent<Rigidbody>().mass +
 offset.magnitude * distanceAlbastra / (distanceAlbastra + distanceVerde);
      greutateAlbastra.GetComponent<Rigidbody>().mass = masaNouaAlbastra;
      greutateVerde.GetComponent<Rigidbody>().mass = masaNouaVerde;
      Debug.Log($"Masa noua greutate albastra: {masaNouaAlbastra}, masa noua greutate
 verde: {masaNouaVerde}");
    }
}
```

The dynamic mass-adjustment algorithm is essential for validating and optimizing the drone's CoG, balance, and stability. Users can directly specify coordinates for a target CoG location, which the algorithm then uses to calculate the necessary mass changes for each component.

Functionality of the Application

The primary functionalities of the application are:

Calculation of the center of gravity: the application calculates the CoG of the drone based on the masses and positions of its components in real-time. This involves computing a weighted average of the positions of the individual components, taking into account their masses.

Real-time mass modification: users can interact with the application to change the mass of individual components and observe how this affects it. The application immediately recalculates and visualizes the updated center as changes are made.

These two functionalities work together to provide users with a dynamic tool for understanding how modifications in mass distribution impact the balance and stability of a complex assembly, such as the drone.

User interface (UI) of the application

The Unity platform was used to create a user-friendly interface that allows for easy interaction with the drone model and the center calculation system. The interface includes the following elements:

Control Panel, which is positioned alongside the 3D window, providing input fields and sliders that allow users to modify the mass of individual components in real-time. Each slider corresponds to a specific part of the drone (e.g. fuselage, arms, propellers, battery) and changes in mass are reflected immediately in the center calculation.

Center of gravity display: it is a section of the UI displays the numeric coordinates of it, showing the X, Y, and Z positions relative to the drone's frame of reference. This helps users monitor how the center changes as they adjust the drone's configuration.

The C# scripts for calculating the center and changing component masses are seamlessly integrated into the Unity environment. The application ties together 3D visualization, user interaction, and mathematical computation in a way that allows users to:

- Modify the drone's configuration dynamically.
- Visualize the impact of those changes on the center in real-time.
- Compare the results with theoretical calculations to validate the accuracy of the method.

4.4. Printing and assembly for the 3D model

In order to determine the center of gravity on a physically complex body, the 3D printer was used to make its components. After 3D modeling with the help of the CATIA program, each part was taken separately to be printed. The material used was PLA. Material fill was between 2% and 100%. The thickness of the walls was adapted according to its need (between 1.2 mm and 5 mm, for the pieces that did not have 100% filling with material). Figure 5 shows the physical model of the drone.

This chapter describes the process of creating and assembling a physical 3D model of a drone to validate the theoretical calculations of the center of gravity. The method involved using a 3D printer to manufacture the individual components based on a detailed 3D model created with CATIA software.

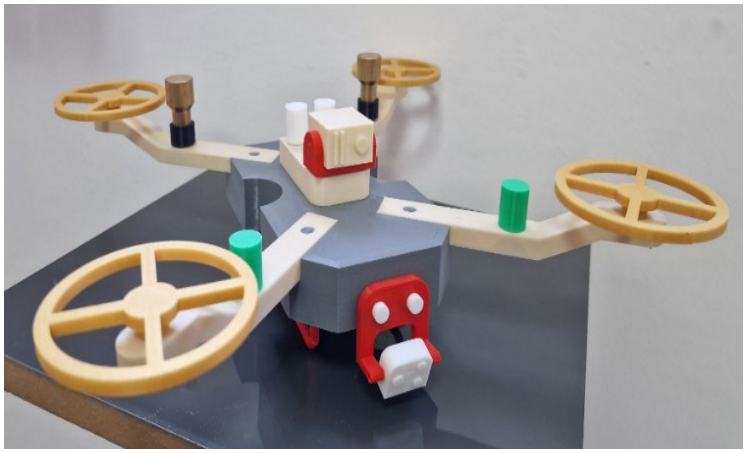

Fig. 5. *Physical model of the complex body*

Assembly of the physical model

Construction - after printing, the individual components were assembled to create the complete drone model. This physical assembly mirrored the 3D model used in the theoretical calculations.

Verification - the assembled drone was then used to verify the center calculations by comparing the theoretical center with the experimentally measured center of the physical model.

With all components printed, the next step was the assembly of the physical drone model. The assembly process was executed with precision to ensure that the final model accurately represented the theoretical design.

- **Component fitting**: each printed part was fitted together according to the design specifications from CATIA. The assembly involved aligning and connecting parts to form the complete drone structure, as shown in Figure 5.
- **Verification**: the assembled model was checked for any misalignments or issues that might affect the accuracy of the center of gravity calculations. Any adjustments were made to ensure that the model was fully functional and ready for testing.

The printing and assembly of the 3D model provided a practical means to validate the theoretical center calculations. By using PLA for the printing process and following precise assembly procedures, the study ensured that the physical model accurately represented the theoretical design.

5. Analytical calculation results

In order to determine the center of gravity of the assembly (*drone*), analytical calculations were carried out. Table 3 shows the results for determining the coordinates on the three axes of each complex body, where X_C, Y_C, Z_C represent the coordinates relative to the system of the complex body, and X_F, Y_F, Z_Frepresent the coordinates relative to the single chosen reference system.

Three axes coordinate for centre of gravity Table 3

Body analysed	X_C	Y_C	Z_C	X_F	Y_F	Z_F
1	8.403997	5.000638	1.495672	0.903997	0.000638	1.495672
2	-3.46042	-2.0055	0.733471	0.039577	-0.0055	3.733471
3	0	0	0	2.5	0	6
4	0	0	0.170198	0	0	5.123198
5	7.60761	1	0.935225	0	0	2.935225
6	-1.47964	0.001709	1	2.52036	0.001749	6
7	0	1	0.545892	1.5	1	5.045892
8	7.60761	1	0.935225	0	0	2.935225
9	0	0	0.170198	0	0	5.123198
10	-0.78336	0	0	9.664642	1	1
11	-0.78336	0	0	9.664642	-1	1
12	0.363599	0	1.804602	10.3636	0	0.804602
13	0.507816	1.24612	0.985186	11.00782	0.00112	-1.26481
14	0	0	0	11	0	-1.25
15	0	0	0.889162	0	0	-2.11084
16	7.60761	1	0.935225	0	0	2.935225
17	0	0	0.889162	0	0	-2.11084
18	0	0	0.170198	0	0	5.123198
19	7.60761	1	0.935225	0	0	2.935225
20	0	0	0.170198	0	0	5.123198

Table 4 shows the final results for the center of gravity of the complex assembly. X_C, Y_C, Z_C represent the coordinates on the three axes of the reference system.

Coordinates of centre of gravity for complex assembly Table 4

Calculation formulas	$Xc = \frac{\sum_{i=1}^{n} Xf * Vi}{\sum_{i=1}^{n} Vi}$	$Yc = \frac{\sum_{i=1}^{n} Yf * Vi}{\sum_{i=1}^{n} Vi}$	$Zc = \frac{\sum_{i=1}^{n} Zf * Vi}{\sum_{i=1}^{n} Vi}$
Final result	0.836354	0.009872	2.195715

6. Validation of analytical results through experimental setup

With the help of the Unity engine, the application was made that can determine the center of gravity of a complex body. In order to observe how the center of gravity changes depending on the mass of the bodies, two buttons have been introduced through which the weight of the bodies positioned on the wings of the drone can be entered manually.

After entering these data, the application will show the exact real-time position of the complex body. Figure 6 shows the application interface for determining complex bodies with variable bodies such as weight.

In this chapter, the focus was on the validation of the analytical calculations for determining the center of gravity by testing the application developed in the previous chapters. The goal was to ensure that the theoretical results obtained from the analytical method align with the experimental measurements of the physical 3D model.

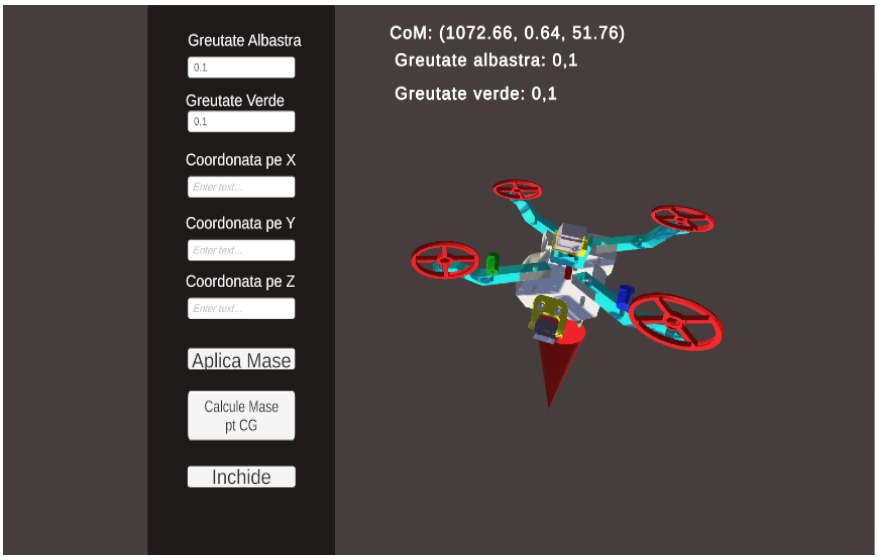

Fig.6. *Interface of the application created*

The results were validated through a physical experiment. The coordinates obtained from the analytical calculations and the developed application were compared to determine the center of gravity of the physical model. A special support was used to keep the assembly in a stable equilibrium position. In Figure 7, the experimental confirmation of the accuracy of the calculations and the application in determining the center of gravity of the assembly is presented.

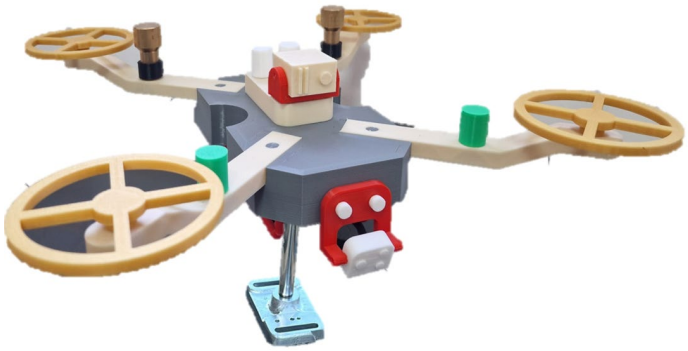

Fig. 7. *Physical validation of the results by experiment*

The validation process demonstrated that the application created for calculating the center and the physical 3D model were both effective tools for verifying the accuracy of the analytical calculations.

By testing both the application interface and the physical model, this chapter provided a comprehensive validation of the center determination process. The successful validation supports the use of the analytical method and the application in real-world engineering applications, highlighting their utility in designing and analyzing complex assemblies.

7. Conclusions and upcoming research

Following the study, we can draw the conclusion that determining the centers of gravity is extremely important in engineering. Also, following the realization of an application with the help of programming in the C# language, the centers of gravity can be determined much faster and precisely. It should be noted that the analytical part helped to create the code and finally to validate the results with the help of the model made after 3D printing.

This study has achieved several key outcomes in the determination of the center of gravity for complex bodies, particularly focusing on a drone model:

- **Effective analytical method**: We developed a reliable analytical method for calculating the center of complex assemblies. This method uses mathematical principles and integrals to ensure accurate results.
- **Successful 3D modeling and printing**: A detailed 3D model of the drone was created and printed using PLA. The model, divided into 20 parts, accurately represented the theoretical design and allowed for effective validation.
- **Functional application**: An application was built using Unity and C# to calculate and visualize the center. Testing confirmed that the application works as intended, providing accurate results based on user inputs.
- **Validation of results**: The physical model was used to verify its calculations. Experimental measurements closely matched the theoretical results, confirming the accuracy of both the analytical method and the application.
- **Practical implications:** The study highlights the importance of combining theoretical calculations with practical tools like 3D printing and software applications. This approach is valuable for engineering fields where balance and stability are crucial.

The following are proposed directions for future work, such as: extension to more complex geometries and bodies, the analysis and determination of CoG to a wide range of geometries and assemblies, that would demonstrate the adaptability and test the calculation and software. Another direction proposed is to develop the software capability, building on the Unity application, which may include real-time updates, based on adjustments, automatic optimization of mass distribution and visualizations of CoG under varying loads.

References

1. Andrea, M., Matteo, Z., Laura F., Domenico, G., Andrea, C., Chiarella, S.: *Validation of a protocol for the estimation of three-dimensional body center of mass kinematics in sport*. In: Gait & Posture, 39(1) (2014), p. 460-465.
2. Colum, D.M., David, A.W.: *Control of whole body balance in the frontal plane during human walking*. In: Journal of Biomechanics, 26(6) (1993), p. 633-644.
3. Ference, M., Weinberg, A. M.: *Center of Gravity and Center of Mass*. In: American Journal of Physics, 6 (1938). https://api.semanticscholar.org/CorpusID:121398880
4. Hansen, C., Rezzoug,N., Gorce, P., Isableu, G., Venture, G.: *Center of pressure based segment inertial parameters validation*. In: Plos One (2017) p. 1-14.
5. Hatze, H.: *A mathematical model for the computational determination of parameter values of anthropomorphic segments*. In: Journal of Biomechanics, 13, (1994), p. 833-843.
6. Marit, P van D., Marco J.M.H., Monique, A.M.B., Veeger, H.E.J.: *From theory to practice: Monitoring mechanical power output during wheelchair field and court sports using inertial measurement units*. In: Journal of Biomechanics, 166 (2024), p. 1-9.
7. Meibao, W., Xiaolin, Z., Wenyan, T., Jun, W.: *A structure for accuracy determining the mass and center of gravity of rigid bodies*. In: Applied Science, 9 (2019), p. 1-13.
8. Mikko, V., Juha, I.: *Determining the location of the body's center of mass for different groups of physically active people*. In: Journal of Biomechanics, (2014), 2532, DOI: 10.3390/app9122532
9. Pufu, E., Moșteanu, A.: *Geometria maselor şi centre de greutate*. [Geometry of Masses and Centers of Gravity]. In: Proceedings of the 11th Conference on Applied and Industrial Mathematics, 2 (2003).
10. Ruilin, Z., Jian W., Sungjun, L.: *Research on the Device of Measuring the Centre of a Weight.* In: Journal of Physics: Conference Series (2021), 1838. DOI: 10.1088/1742-6596/1838/1/012072
11. Ruilin, Z., Jin'an, D., Jian, W.: *Investigation of two methods of volume measurements based on hydrostatic comparison method*. In: IOP Conf. Series: Journal of Physics, 1065, (2018), p. 1-5.
12. Sfetcu, N.: *Centrul de masă și de greutate*. [Center of mass and weight] In: Revista Mecanica, 2018.
13. Yuancheng, J., Winter, D.A., Ishac, M.G., Gilchrist, L.: *Trajectory of the body COG and COP during initiation and determination of gait*. In: Gait & Posture, 1 (1993), p. 9-22.
14. https://users.utcluj.ro/~tudor_milchis/data//util/curs%206_draft_01.pdf. Accessed: 10.08.2024.
15. https://warbletoncouncil.org/centro-de-gravedad-4467. Accessed: 08.08.2024