A Vision and GPS Based System for Autonomous Vertical Landing of UAV Quadcopter

A S Priambodo[1], F Arifin[2], A Nasuha[3], Muslikhin[4], A Winursito[5]

[1,2,3,4,5]Department of Electronics and Informatics Engineering of Education, Engineering Faculty, Universitas Negeri Yogyakarta, Indonesia

E-mail: `ardyseto@uny.ac.id`

Abstract. All articles *must* contain an abstract. This document describes the preparation of a conference paper to be published in *Journal of Physics: Conference Series* using LaTeX 2_ε and the `jpconf.cls` class file. The abstract text should be formatted using 10 point font and indented 25 mm from the left margin. Leave 10 mm space after the abstract before you begin the main text of your article. The text of your article should start on the same page as the abstract. The abstract follows the addresses and should give readers concise information about the content of the article and indicate the main results obtained and conclusions drawn. As the abstract is not part of the text it should be complete in itself; no table numbers, figure numbers, references or displayed mathematical expressions should be included. It should be suitable for direct inclusion in abstracting services and should not normally exceed 200 words. The abstract should generally be restricted to a single paragraph. Since contemporary information-retrieval systems rely heavily on the content of titles and abstracts to identify relevant articles in literature searches, great care should be taken in constructing both.

1. Introduction

Unmanned Aerial Vehicle (UAVs) or Drones is a topic of great interest to researchers now. A drone can do much work. In agriculture, drones are used as sprinklers of liquid fertilizer and observation of crops from the air[1]. Drones are used as spies from the air in the military field[2]. In the safety field, drones are used for surveillance in disaster areas[3]. The quadcopter is one type of UAV that is widely developed because it has the advantage of being able to take off and land vertically. The quadcopter consists of four rotors that rotate the propellers to get lifts. Because the quadcopter can take off and land vertically, the required take-off area is minimal. In addition to these advantages, the quadcopter also can hover in the air.

There are various types of UAVs, namely fixed wing, rotary wing and hybrid fixed and rotary wing. In comparison, the rotary wing consists of 2 types: single rotor and multirotor. The hybrid and rotary wing UAV can take off and land vertically, so they do not require a large area. The selection of the type of UAV depends on the task and area to be completed. Multirotor has many advantages such as more agile manoeuvrability, ability to hover in the air, with a certain size, it can fly indoors, take off and land vertically and has a relatively simple frame design compared to other types. Multirotor also has disadvantages, namely limited flight time, which causes relatively small cruising area and limited carrying capacity. The multirotor that is widely developed is the quadcopter or quadrotor type. The quadcopter is a multirotor with 4 motors driving the propellers to generate lift. The quadcopter can be configured to be type X or H.

In general, flying vehicles' flight actions are based on GPS sensors. A global positioning system (GPS) is an electronic device that can be used for navigation. GPS provides latitude, longitude and altitude values as a reference from the UAV so that the UAV can fly to a certain point based on that reference. GPS on UAVs, especially quadcopters, is also used to get the take-off location and later used as a landing point. However, the GPS does not provide accurate values in some conditions and environments, so the landing becomes imprecise. In limited applications where the take-off and landing places are very narrow, this is a challenge, so we cannot just rely on GPS to find out the location of the landing point.

Manually landing a quadcopter usually has difficulties. For example, pilots find it difficult to gradually reduce the quadcopter's altitude and land smoothly and accurately at precise locations. Automatic landing using GPS is a method to facilitate the quadcopter land smoothly and accurately. However, in particular area and time conditions, the GPS accurateness becomes less good and causes the quadcopter not to land in the accurately determined area[4]. In this paper, we propose a new automatic landing method based on GPS and computer vision so that the quadcopter can land precisely in the specified area. The take-off and landing area used in this study is an ArUco marker. Improved landing accuracy, safety and reliability are the advantages of our proposed method.

2. Methodology
2.1. Quadcopter Model
A quadcopter is a type of flying vehicle which has four motors used to rotate the propeller. The propeller's rotation here will generate lift to raise the quadcopter. Adjusting the speed on each quadcopter motor can perform roll, pitch, yaw or vertical movements. Quadcopters with X configuration consist of front right, rear right, rear left, and front left motors, usually numbered 1,2,3 and 4. Quadcopters with an X configuration of 1,3 and 2,4 propeller pairs rotate in opposite directions, eliminating momentum and not requiring a tail rotor for counterbalance. Besides, this research uses an X configuration quadcopter because it has better stability than the + configuration.

The roll movement is influenced by the speed of the right part (1,2) and left part(3,4) motors. Pitch movement is affected by the speed of the front part (1,4) and rear part (2,3) motors. Yaw movement is influenced by combination of motors 1,3 and 2,4. The speed of the four motors on the quadcopter influences the vertical movement. The illustration of the quadcopter movement can be seen in Figure 1.
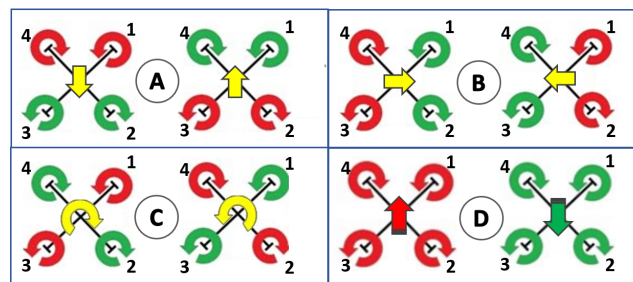
Figure 1. Quadcopter Basic Movement: (A)Pitch (B)Roll (C)Yaw (D)Vertical Movement (the red color means faster than green)

Based on rotational and translational movements, a quadcopter mathematical model can be derived for 6-DOF using the Newton-Euler technique. Using this approach, the 6-DOF were deduced show in equation 1-6.

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{J_r}{I_x}\dot{\theta}\Omega + \frac{l}{I_x}U_2 \tag{1}$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{J_r}{I_y}\dot{\phi}\Omega + \frac{l}{I_y}U_3 \tag{2}$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) + \frac{l}{I_z}U_4 \tag{3}$$

$$\ddot{x} = (cos\phi sin\theta cos\psi + sin\phi sin\psi)\frac{U_1}{m} \tag{4}$$

$$\ddot{y} = (cos\phi sin\theta cos\psi - sin\phi sin\psi)\frac{U_1}{m} \tag{5}$$

$$\ddot{z} = -g + (cos\phi cos\theta)\frac{U_1}{m} \tag{6}$$

2.2. Vision-Based Landing

Several artificial fiducial marking systems exist April-Tags, ArUco, and ARToolKit. With this comparison, ArUco detection can detect markers quickly and reliably. Therefore, the ArUco marker was used as a landing pad in this study. The ArUco marker is a synthetic square marker composed of a wide black border and an inner binary matrix that defines its identifier. Black border for fast image detection and binary codification enables and application of detection and error techniques. The marker's size determines the internal matrix's size; for example, the 4x4 marker size consists of 16 bits. ArUco markers generated for landing pads have specifications, as shown in Table 1. We can create ArUco markers using python and OpenCV or through websites that provide ArUco Generator services. The generated marker is an image file in PNG or JPG format shows in Figure 2.

Table 1. Parameter of ArUco marker used

Parameter	Value
Dictionary	6x6 (50, 100, 250, 1000)
Marker ID	1
Marker size (mm)	100

By using OpenCV in the ArUco marker detection process, you will get two values, namely (1) the position of the four corners of the marker; (2) the id of the marker. In this study, what is needed is the position of the four corners so that the midpoint of the marker can be calculated, which can then be calculated for the error in the position of the camera's centre. The marker detection process consists of several steps. The first step is to apply an adaptive threshold by calculating the average value of the surrounding pixels of a given pixel. This way, all the contours in the image are expected to be found, and then the non-square shapes are filtered. The next step is to extract the marker bits in the picture. The first perspective transformation is applied to get the marker in canonical form. Then the canonical image is subjected to a threshold process using otsu to separate the white and black bits. The image is divided into cells according to marker size and border size.
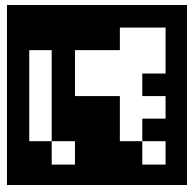

Figure 2. ArUco marker used in this research

The number of black and white pixels in each cell is then counted to determine whether it includes black or white bits. The final step is to compare with a specific dictionary. The results obtained are the position of the four corners and the id value of the marker. After getting the position of the four corners of the marker, the centre of the marker can be calculated, which can later be used to calculate the position of the X and Y errors by comparing with the midpoint of the camera capture, as shown in Figure 3. The X and Y position error values will be used to adjust the roll and pitch angle of the quadcopter to correct its position according to the desired set point.
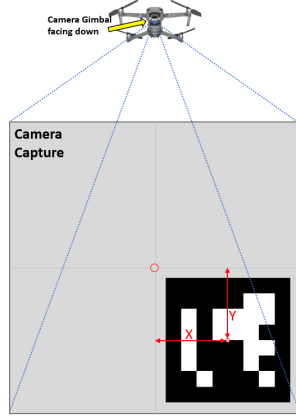
Figure 3. Calculation X and Y position error based on detection of ArUco Marker

2.3. Proposed Landing Control

GPS is a Global Positioning System, a tool or system that can use to inform users that they are (globally) on the surface of the earth based on satellites. GPS is installed on the flight controller so the flight controller can find out where the quadcopter is. The flight controller can store the take-off position so that, theoretically, the quadcopter can land in the take-off area. In addition, GPS can also function as flying navigation at specific coordinate points.
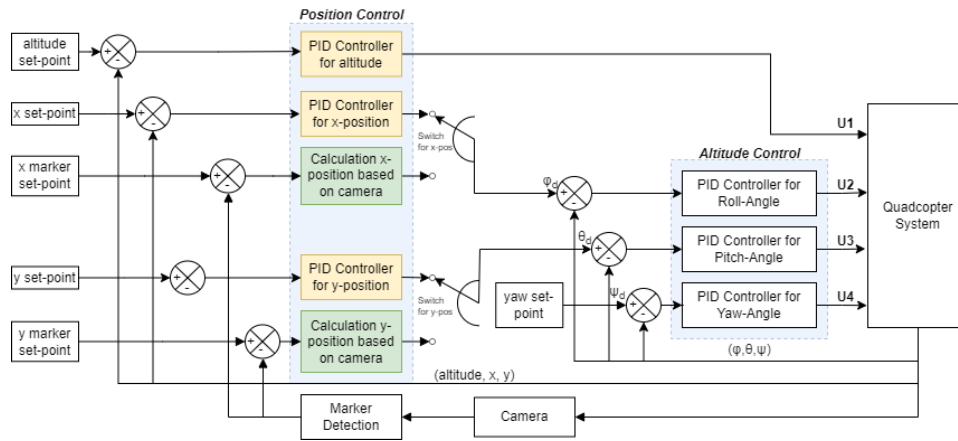
Figure 4. Block Diagram of Proposed Controller

There are two control loops designed, namely attitude control and position control. Position control is in charge of adjusting the translational position of the quadcopter so that it is right at specific coordinates and altitudes. The output of the position control is the set point values for the roll and pitch angles. The roll and pitch set point values will be input for attitude control so that the quadcopter will move to follow the set points from a predetermined position. The block diagram used in this control proposal is shown in Figure 4. The control system used in this proposal is a PID control system known to be reliable in correcting various systems. The PID

control system consists of 3 components: proportional control, integral control, and derivative control. The equation of the PID control system is shown in equation 7.

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{7}$$

It has been mentioned that GPS has shortcomings, so there is an additional camera used to detect the ArUco marker, which functions as a landing pad. The reference from the position control will switch from GPS to the estimated position based on the camera when the quadcopter camera detects the marker. That way, the quadcopter will position exactly above the marker, and the process of lowering the height is carried out slowly until the quadcopter touches the ground. The proposed control running the algorithm is shown in Figure 5.
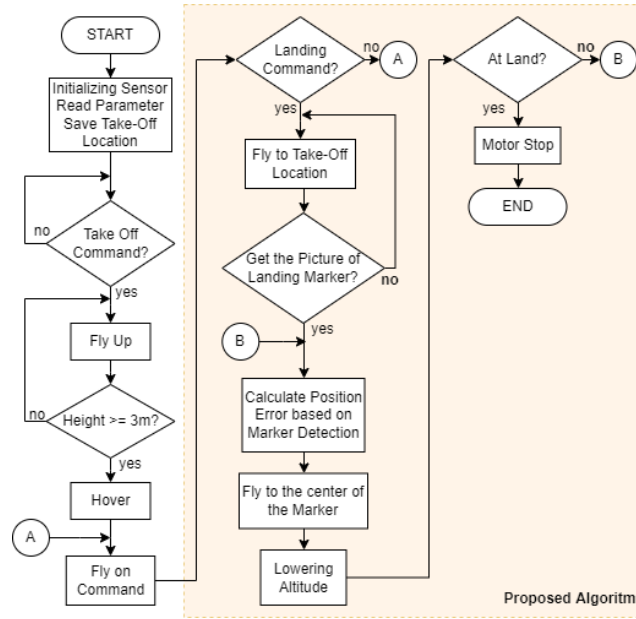
Figure 5. Proposed Flowchart of Vision and GPS Based System for Vertical Landing Quadcopter

2.4. Simulation Environment

Webots is an application or software developed by the Swiss Federal Institute of Technology that serves to create models, programs, and simulations of a robot. Using webots, we can design or design and develop programs that will run robots in an artificial environment. This application was first developed in 1996 by Dr Oliver Michael. Then in 1998, it was developed again by Cyberbotics Ltd. as proprietary licensed software. Thanks to Cyberbotics Ltd in 2018, made webots an open source application so that it is easy to use and its development becomes more available. Webots can be installed on Windows, Mac OS and Linux (Ubuntu) operating systems.

Webots provides many components in the form of robots, sensors, actuators and other objects such as humans, animals, boxes and many other things that we can use in an artificial environment. In addition, we can also create our objects through 3D CAD applications.The programming languages supported by webots also vary, namely Matlab, C/C++, Python, Java and ROS. This study uses a DJI Mavic 2 Pro drone shows in Figure 6, in which the proposed controller will simulate a yard environment consisting of several objects such as houses, cars, roads, and many trees. Webots version used in theis research is R2022a. The landing pad used to search at the top is displayed with the ArUco marker, which the quadcopter will later recognize as the area for landing. DJI Mavic 2 Pro consists of several sensors such as an accelerometer,

gyroscope, barometer, compass and GPS that we can access through the program. The webots simulation program is run on a computer with the specifications listed in Table 2.

Table 2. Computer Spesification

Component	Description
CPU	Intel i5 12400F
RAM	2x8GBs
GPU	RTX 3060 12GB
Storage	SSD NVMe Gen4x
Operating System	Ubuntu 20.04
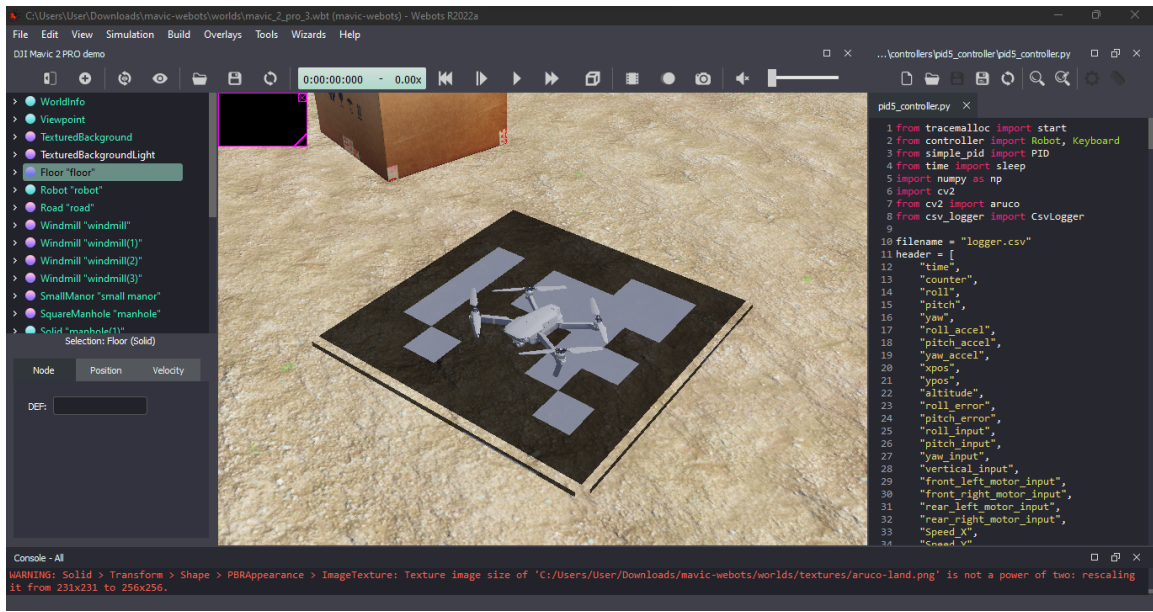
Figure 6. DJI Mavic 2 Pro in Webots Environment

3. Result and Discussion

This section will show the results of testing the proposed control system—several tests on the ArUco marker detection section and overall system testing involving GPS and Computer Vision.

3.1. Marker Detection

The ArUco marker detection test is carried out by positioning the quadcopter at several X and Y positions and heights, as shown in Figure 7. To make it easier to observe, the roll, pitch, and yaw angle data, as well as the X, Y, and altitude positions, are displayed on the simulation screen of the webots. Before testing, the gimbal camera must be directed downwards, and there is a control mechanism that makes the camera direction always perpendicular to the ground so that it is hoped that the marker is caught. The camera is in a perfect square shape. Obtained at the height of 2.17 meters is the minimum height that the quadcopter must reach for the camera to detect the presence of a marker. In comparison, the maximum value is 29.85 meters. Beyond this altitude value, the marker cannot be detected, so the quadcopter is set when returning to home mode at 20 meters. The results of this height measurement are shown in Figures 8 and 9, where the marker will be overlaid with a transparent green box.
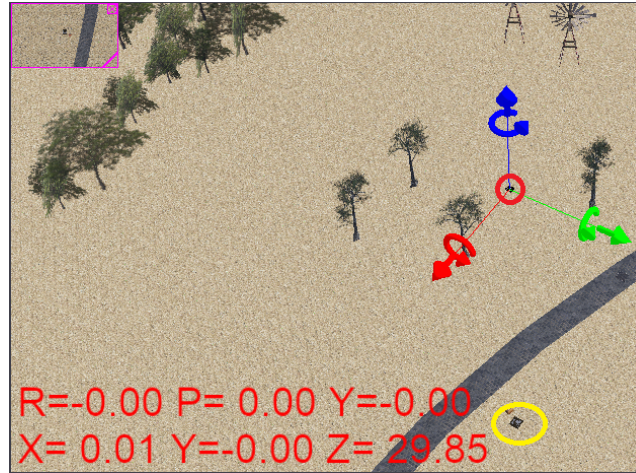
Figure 7. Marker detecion testing in Webots. There is information about the translational and rotational position of the quadcopter which is displayed in red font. The quadcopter is marked with a red circle and the ArUco marker is marked with an yellow circle.
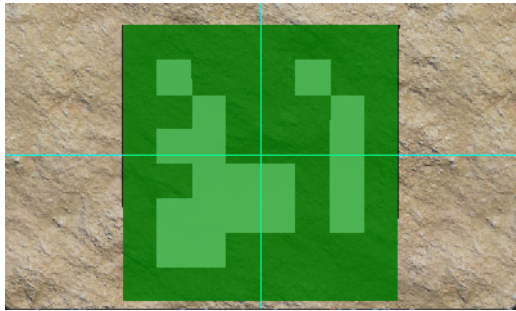
Figure 8. Minimum range of ArUco marker detection at a distance of 2.17 meters
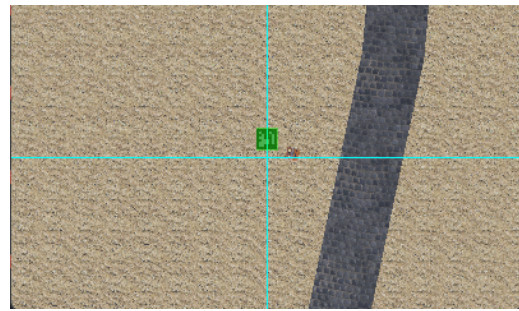
Figure 9. Maximal range of ArUco marker detection at a distance of 29.85 meters

3.2. Precision Landing
test test

4. Conclusion
test test

References
[1] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BIBT_EXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).
[2] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BIBT_EXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[3] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[4] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[5] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[6] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[7] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[8] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[9] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[10] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[11] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[12] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[13] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[14] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[15] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).

[16] IOP Publishing is to grateful Mark A Caprio, Center for Theoretical Physics, Yale University, for permission to include the `iopart-num` BibTEXpackage (version 2.0, December 21, 2006) with this documentation. Updates and new releases of `iopart-num` can be found on `www.ctan.org` (CTAN).