Baran Usluel

Extended Essay

Physics

“To what extent does the number of degrees of freedom in an articulated robot manipulator influence the payload, repeatability and work envelope?”

Word Count: 3897

Abstract

The objective of this project was to understand and model the relationship between the number of degrees of freedom in an articulated robot and three factors: the payload, repeatability, and work envelope. Currently, industries looking to implement robots into their production lines and businesses must make a number of decisions when choosing what type of robots to utilize. One of the most notable factors to be decided upon is the number of degrees of freedom, as this has a pronounced effect on many of the robot's other characteristics. While some specialized tasks (such as lifting exceptionally heavy objects) may be better suited by robots with an uncommon and more extreme number of degrees of freedom, most tasks require well-rounded robots with balanced characteristics. That is why the models developed in this investigation were used to identify how many degrees of freedom are best for the latter type of robot.

The payload model was derived using torque calculations. The repeatability model was calculated by applying the root-sum-squared method on the uncertainties introduced by each of the motors, extrapolated to the robot's end point. And finally, the work envelope was evaluated by a MATLAB program written by the author, which uses an iterative approach along with the concept of robotic forward kinematics to determine the number of unique and total points that can be reached by the robot.

The ideal number of degrees of freedom was found to be 5, by numerically approximating the intersection of the increasing and decreasing functions of the degrees of freedom. This value matched that of commonly used industrial robots, indicating that industries are indeed using the most balanced and optimal robots.

Word Count: 275

Table of Contents

Abstract i

Table of Contents ii

Introduction 1

Research Topic 1

Theories 1

Definition of Terms 2

Thesis 6

Method 7

Calculations and Simulations 14

Payload 14

Repeatability 16

Work Envelope 18

Analysis 20

Conclusion 23

Works Cited 24

Appendix 25

MATLAB Source Code for Work Envelope Calculations 25

Introduction

Research Topic

> *To what extent does the number of degrees of freedom in an articulated robot manipulator influence the payload, repeatability and work envelope?*

The purpose of this extended essay is to investigate the relationship between the number of degrees of freedom in an articulated robot manipulator (more commonly known as a robot arm), and its three main characteristics, which are the payload, repeatability, and work envelope.

Once the necessary models are derived, they will be used to determine what number of degrees of freedom will yield the most balanced robot, in terms of the various characteristics. This means that the robot will neither be incredibly strong but imprecise, nor precise but inflexible; instead, the most uniform and optimal value will be found. Finally, the resultant value will be compared with the number of degrees of freedom found in the most popular industrial robots, to establish whether those robots are also balanced, or more specialized in certain areas.

Theories

This research paper on articulated robot arms will make use of several concepts. Firstly, the evaluation of the work envelope variable will involve robot kinematics, which is the “analytical study of the motion of a robot manipulator” (Kucuk and Bingul). More specifically, forward robot kinematics will be used, where each of the joint variables / angles are taken in as input, and the position and orientation of the end-effector (end of the robot arm) is calculated.

In addition, this paper will make use of the Physics concepts of torque and rotational dynamics, when calculating the payload capabilities of the various robot arms.

And finally, geometry and trigonometry will be used, both in the derivation of the forward kinematics solution, and also in the repeatability calculations.

Definition of Terms

Robot Manipulators

Robot manipulators are programmable machines used to move materials, parts and tools to perform certain tasks. Robots are particularly beneficial when used to perform unsafe, repetitive or unpleasant tasks. They can be used for material handling, assembly, welding, painting, etc. (United States Department of Labor).

Degree of Freedom (DOF)

A degree of freedom is a single independent direction of motion in a robot manipulator. There are three main types of degrees of freedom in robots. The first is rotation about an axis parallel to the arm, like the human wrist. The second is linear movement, in which a component slides in and out, or up and down, and so forth. The third is rotation about an axis perpendicular to the arm, like the human elbow (Vex Robotics).

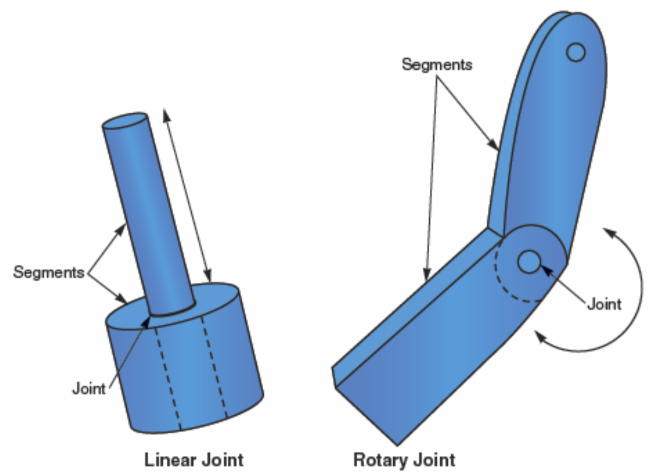


Figure 1 (Ross, Fardo and Masterson)

Motors

The most commonly used type of motor in robot arms is the servo motor. Industrial servo motors have sensors that monitor position and velocity in the real-time, and they adjust their motion based on a feedback loop. They can provide very high torques and are ideal for most applications. Hobby servo motors are affordable and relatively light motors that can be used for small-scale prototyping, but are not as powerful or efficient.

Payload

"A robot payload is the maximum of weight that a robot can pick up or manipulate" (RobotWorx). The payload of a robot arm depends on various factors, such as the design of the arm, the masses of the components, and the torque specifications of the motors. It is one of the most important properties of a robot, as a low payload can limit the potential uses and functions of a robot.

Work Envelope

"A robot's work envelope is its range of movement. It is the shape created when a manipulator reaches forward, backward, up and down" (RobotWorx). Simply put, the work envelope is the volume of space that the robot's end-effector can reach. The work envelope depends on the robot's reach length, the number of DOF, and the design of its axes (joint configuration). "Many of the robots are designed with considerable flexibility. Some have the ability to reach behind themselves" (RobotWorx). This is a beneficial ability, because it means that the robot is less constrained and can make more difficult maneuvers.

Repeatability

"[Repeatability] is the ability of a robot to return to the same position" (RobotWorx). This is crucial for a robot, especially in an industrial setting, because poorer repeatability results in lower quality work and products. The repeatability of a robot can be improved through the use of sensors and feedback loops, but implementing these requires additional resources.

Types of Robot Arms

Robot arms are classified in accordance with their design configurations. The different configurations have a large influence on the work envelope, payload, and repeatability. Three common configurations will be discussed:

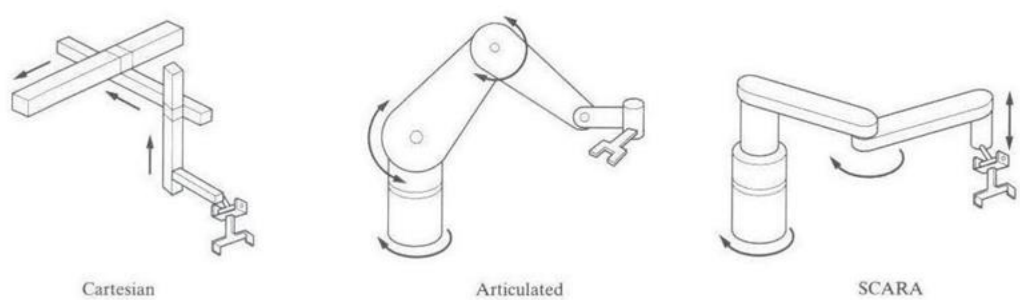


Figure 2 (Adapted from Wysk)

The first configuration is called the rectangular coordinate robot. It consists of three linear joints that move along the Cartesian coordinate system. It is ideal for manipulating high payloads, due to its rigid structure. This configuration has little or no rotary capabilities, limiting its work envelope and deeming it unsuitable for applications that require complex and flexible motion (Robot Arm Configurations).

The next one is the SCARA arm. It is made up of multiple rotary joints that all rotate around the vertical axis, and a single vertical linear joint. It has the advantage of being vertically rigid while retaining flexibility in its horizontal motions. Nonetheless, it also has a limited work envelope and is often not preferred (Robot Arm Configurations).

The most common robot arm configuration is the revolute configuration. Also known as an articulated arm, it only consists of rotary joints. It is said to be anthropomorphic because its movements and joints resemble those of the human arm (Robot Arm Configurations). Many industrial articulated arms will have around 6 DOF, granting them complete versatility (Ross, Fardo and Masterson). This configuration yields considerable flexibility in regards to motion, but it is not as rigid as the other configurations.

This research paper will be focusing on articulated robot arms, for two primary reasons: they are the most common, especially in industrial settings; and they are particularly prone to payload issues, which this research aims to investigate and minimize. Unconventional, experimental designs won’t be considered here, for the sake of simplicity and consistency. Instead, the investigated robots will be modeled after commonly used industrial robot arms like those made by Yaskawa Motoman Robotics, and other large automation companies.

Thesis

Research Topic:

> *To what extent does the number of DOF in an articulated robot manipulator influence the payload, repeatability and work envelope?*

Increasing the number of DOF results in a larger number of motors, meaning that the mass of the arm itself will be greater. Additionally, each new DOF will introduce another potentially vulnerable joint to the system, resulting in a deterioration of rigidity. This means that the payload will decrease with each additional DOF, and the robot will not be able to manipulate heavy loads.

Similarly, the repeatability of the movements will decrease because of the tolerances of the individual motors, as well as the backlash on every joint. The uncertainty that is introduced with each new motor, when extrapolated to the end-effector, may cause significant repeatability limitations.

On the other hand, increasing the number of DOF will enlarge the work envelope (even though the reach of the arm is constant) and give greater flexibility to the arm (such as reaching a certain point from various angles and sides). This can be particularly valuable in industrial settings where the arm is expected to conduct complex movements and maneuvers.

Based on the specifications of existing commercial robot arms, it can be expected that the ideal number of DOF for typical applications will be near 5 or 6 (RobotWorx). This will certainly differ from one application to the next, but it should be possible to determine a so-called ‘sweet-spot’ where the trade-offs are most balanced.

Method

The experimental research will consist of making calculations and running simulations. Equations describing various factors will be derived, and simulation results will be analyzed. The expressions for the dependent variables will be used to calculate the optimal DOF value. The result will then be compared to the hypothesis, that the value should be 5 or 6.

Independent Variable

The independent variable will be the number of DOF in the robot, ranging from 1 to 9.

Dependent Variables

The dependent variables will be the payload, repeatability, and work envelope.

In addition to calculating the volume of the work envelope, the flexibility of the motion within that envelope will also be considered.

Control Variables

As it is difficult to find technical specifications for the individual components (i.e. motors, segments) of many real-life industrial robots, the ones being considered here will be at the hobby scale. That is why the robots will have MG 996R hobby servos for all of their joints.

Similarly, all the robots will have a reach length of 30 centimeters, which is very small for an industrial robot but fitting for a hobby robot with the given motor.

External sensors, feedback loops or PID (proportional-integral-derivative) control systems will not be considered, as they are all error-correction mechanisms which would defeat the purpose of the investigation, which is to find the direct relationship between the number of DOF and factors such as repeatability.

Payload

When making payload calculations, it will be assumed that the arm is in its horizontal reach position, as that is the worst-case scenario in which the motors experience the greatest torque.

There will be two notable assumptions in the payload calculations. Firstly, it will be assumed that the center of mass of the item being held by the manipulator is at the precise location of the end-effector. This assumption must be explicitly stated because if the item is large or has an irregular shape, then that can affect the torque of the item and hence the payload of the robot. And secondly, it will be assumed that the mass of the robot arm is distributed evenly along the length of the arm. This isn't the case for most robots, as their structures often get thinner and lighter towards the end-effector, but this assumption will provide a convenient approximation to observe a correlation.

Work Envelope

To calculate numerical values for the work envelopes of the robots, forward kinematics will be evaluated with MATLAB code. An iterative solution will be used, in which each of the joint angles are repeatedly incremented by a small angle, so that they sweep from their minimum to their maximum positions. For each of these angle combinations, the forward kinematics will be calculated to find the end-effector's position. Hence, every single point that the robot arm can reach (at a certain joint precision) will be collected.

However, this alone will not suffice. For example, if one were to use this process on a 1 DOF robot, and then do the same for a up-scaled version of the same 1 DOF robot, the number of points calculated would be the same, but the work envelope for the latter should clearly be larger, indicating that this representation can be flawed. Therefore, the coordinates of each of the calculated points will be rounded to a certain value so that they are all aligned in a 3D grid. This will yield the number of unique points reached, which is indeed representative of the volume of the work envelope.

In addition to the number of unique points, the total number of points reached will also be considered. This is because when there are more degrees of freedom, the same points will be able to be reached in more ways, meaning that the robots will have greater flexibility; and the total number of points reached by the robots will be a fair representation of that. This would be a meaningless measure if the robots had varying reach lengths, but as they are constant (30 centimeters), this variable can be used.

The output of the MATLAB program is visually represented by Figure 3. The indicated points are all the coordinates that a 3 DOF robot can reach. The colors correspond to values of $\log_{10}(V)$, where V is the number of different ways that the robot can reach the given point.

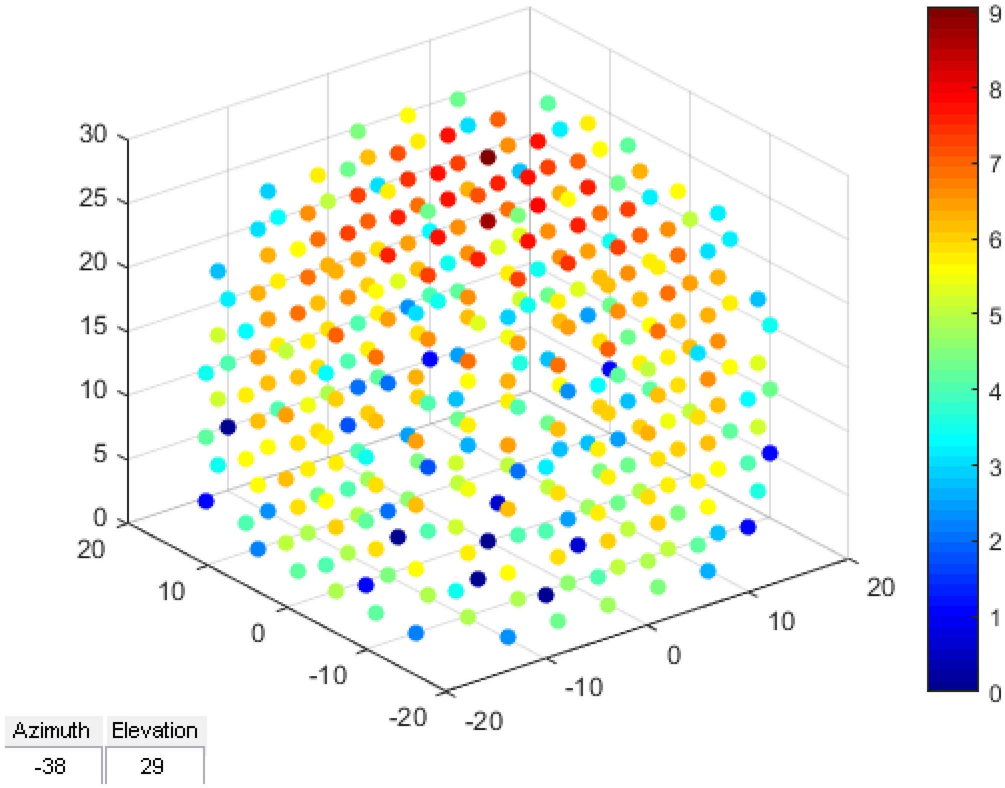


Figure 3, Work envelope graph generated by MATLAB program

Repeatability

The repeatability of the robots will be calculated by geometrically sweeping out the angle tolerances of the motors / encoders at each of the joints to the end-effector's position, and finding the root-sum-squared. It will be expressed as ±x where x is a value in centimeters.

Robot Configurations

Nine different robot manipulators, with DOF ranging from 1 to 9, will be studied. These will all have the same reach length of 30 centimeters, for consistency in the work envelope, payload and repeatability calculations. Industrial robots typically have either 5 or 6 DOF, but ones with 4 or 7 can also be found. The robots in this paper that have such many DOF will be designed similarly to standard industrial robot designs. The other robots will have their joints added and removed in a manner that matches the trend (see Figure 3 below). Note that despite the appearance of varying reach lengths in the figure, it will be kept constant.

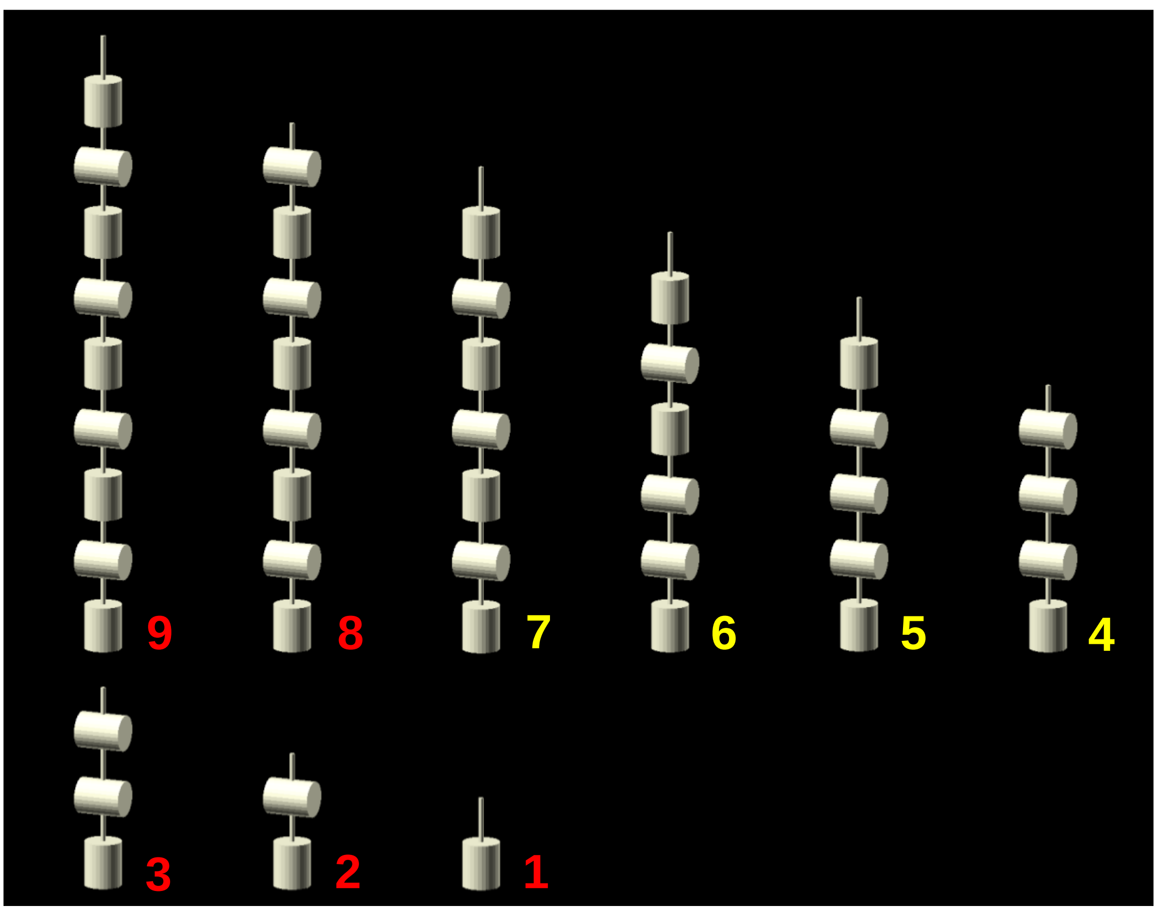


Figure 4, Diagram of joint configurations for DOF from 9 to 1 (where 7 to 4 are standard configurations), rendered using OpenSCAD

Motors

The motors of the robot manipulators being considered will be modelled after MG 996R hobby servo motors. These are small and inexpensive motors, weighing 57 grams, that are typically used for prototyping projects.

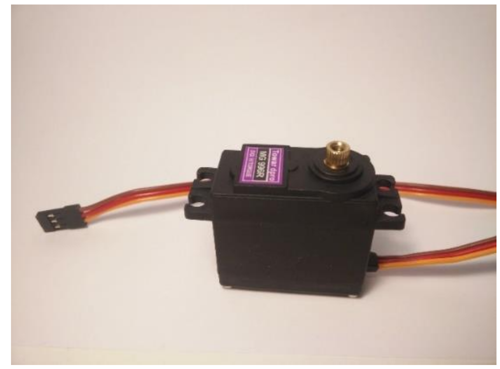


Figure 5, MG996R hobby servo motor

The MG 996R servos are reported to have a stall torque rating of 15kg-cm when powered at 6 volts (MG996R Digital Servo Metal Gear). This value was confirmed experimentally by attaching a 1.5kg load to the motor at 10cm from its axis of rotation; the motor held the load in place, but was not able to move it upwards, as this is only the stall torque. The motor wasn't tested with heavier loads, as it appeared to be under strain, and further tests could have caused physical damage.

The repeatability/resolution of the servo motor was not specified by the seller. It was determined experimentally, by attaching a link (a pen) to the motor to create a simple 1 DOF robot, as shown in Figure 6. This small robot had a reach of 121mm. It was programmed so that it would repeatedly rotate to various positions, and then attempt to return to a target position, where the offset error from the desired position would be measured. By considering the maximum errors, a resolution of ±7mm was found for this specific 1 DOF robot. This value was then used to calculate the angular resolution of the motor as ±3.3 degrees.

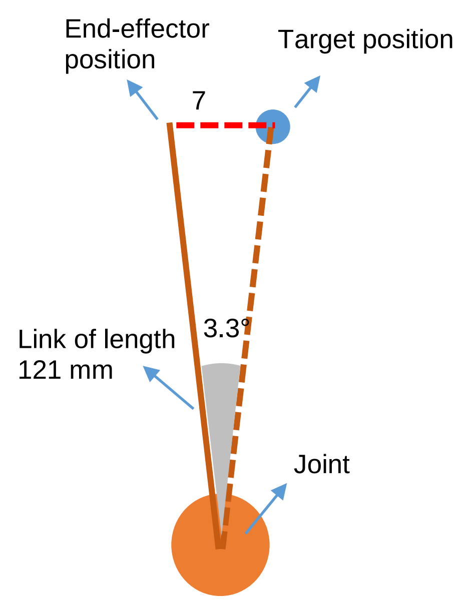


Figure 6, Diagram of repeatability measurement setup

And finally, the servo's range of motion was experimentally determined to be 180 degrees.

Assumptions

The assumptions being made in this investigation are outlined below:

- The center of mass of the item being held by the manipulator is at the precise location of the end-effector. Without this assumption, it would be impossible to perform the torque calculations, as the load's shape (and hence location of center of mass) affects the torque.
- The mass of the robot arm is distributed evenly along the length of the arm. This is a convenient approximation made to avoid trivially asserted masses.
- The same motor is used for each joint of every robot. This assumption is for consistency.
- The total reach length of the robots is 30 centimeters. This assumption is also for consistency.
- The individual link lengths in a single arm are equal to each other (equal to reach/DOF). Once again, this is for consistency and simplicity.
- The unique and total number of points that can be reached by the robot, rounded to a certain 3D grid, is a fair representative of the work envelope. This assumption allows us to model the work envelope in terms of the points reached, as doing so otherwise would be much more difficult.

Calculations and Simulations

Payload

When calculating the payload capacities of the robots, the torque specification of the bottommost motor, the length and the mass of the arm are considered.

The motors weigh 57 ±1 grams and have a torque specification of 15kg-cm. Hobby-scale robots are typically 3D printed, so if the links were to be 3D printed using PLA plastic (density of $1.25g/cm^3$), with an infill density of 25%, and were to have a cross-sectional area of $2cm*2cm=4cm^2$, then the mass of the arm without the motors should be 1.25*0.25*(4*30) = 37.5 grams.

In the first robot, with 1 DOF, the axis of rotation is vertical, meaning that this joint doesn’t have to apply a force to resist gravity. As long as the structural integrity of the robot is sufficient, this robot won’t have any payload limitations. The payload can therefore be considered as infinite or an arbitrarily large value.

The second joint of the robot with 2 DOF has to hold up the arm and the load against gravity. The torque of the arm on the second motor can be shown as (15/2)*(37.5/2) g-cm because the second link composes half the mass of the whole 3D printed arm structure, and its center of mass is at the center of the link, which is 15/2 cm away from the motor. The torque expression can be used to calculate the payload:

$\tau = (15/2)*(37.5/2) + 15*P = 15*1000$ $\qquad$ $P = 990.625$ grams

Similarly, in the 3 DOF robot, the second joint must be considered again because it is the one that will be subjected to the greatest torque. This time, the torque expression will also include the mass of the third motor, and can be shown by:

$\tau = (20/2)*(37.5*(2/3)) + (20/2)*57 + 20*P = 15*1000$ P = 709 grams

Following the same reasoning, the payload can be generalized as:

$$P = \frac{15*1000 - \frac{30-\frac{30}{n}}{2} * \left[37.5*\left(1-\frac{1}{n}\right) + 57*(n-2)\right]}{30 - 30/n}$$

With this equation, a table of DOF vs payload can be generated:

DOF	Payload
1	N/A
2	990 g
3	710 g
4	600 g
5	520 g
6	470 g
7	420 g
8	380 g
9	350 g

Table 1, Effect of DOF on Payload

Repeatability

The repeatability of the 1 DOF robot can be calculated easily. As it only has one joint, and one link of length 30 cm, the angular repeatability (±3.3°) can be extrapolated to the end-effector as shown:

$r = 2*\pi*30*(3.3/360) = 1.7$ cm

Repeatability = ±1.7cm

Note that in the above expression, the repeatability was considered as an arc, instead of a straight line, from the target position to the maximum error position. This approximation may result in very small differences, but they will be negligible.

For the next robot, two joints and two respective links of length 15cm must be considered. If these two joints rotated in the same direction, they could have just been added; however, as they move the end-effector in perpendicular directions, the Pythagorean theorem will be used instead, by calculating the root-sum-squared:

$r = \sqrt{((30*k)^2 + (15*k)^2)} = 1.9$

Repeatability = ±1.9cm

Note that the substitution $k = 2*\pi*(3.3/360)$ has been made for simplicity.

As more degrees of freedom are added, the joint configurations of the robots get increasingly complex. That is why for the sake of consistency, all of the following repeatability values will also be calculated using the simple root-sum-squared method.

In the robot with 3 DOF, the repeatability can therefore be shown as:

$r = \sqrt{((30*k)^2 + (20*k)^2 + (10*k)^2)} = 2.2$

Repeatability = ±2.2cm

It follows that the repeatability of the 4 DOF robot is:

r = √((30*k)^2 + (22.5*k)^2 + (15*k)^2 + (7.5*k)^2) = 2.4

Repeatability = ±2.4cm

The same root-sum-squared method is applied to the next robots, and a table of DOF vs repeatability is constructed:

DOF	Repeatability
1	±1.7cm
2	±1.9cm
3	±2.2cm
4	±2.4cm
5	±2.6cm
6	±2.7cm
7	±2.9cm
8	±3.1cm
9	±3.2cm

Table 2, Effect of DOF on Repeatability

Work Envelope

The work envelopes of the robots were evaluated by a program written by the author in MATLAB, based on the concept of robot forward kinematics (see appendix). This program returns two numbers: the first gives the number of unique points that the robots can reach, where the each of the coordinates are rounded to the nearest 5; the second number gives the total number of points that the robot can reach.

The program was executed with its parameters set to a reach length of 30, angle increments of 0.06 radians, and a maximum angle of pi/2 from the zero position. The data can be seen below:

DOF	Unique Points	Total Points
1	1	54
2	93	2,862
3	342	151,686
4	490	8,039,358
5	605	426,085,974
6	743	22,582,556,622
7	N/A	**1,196,875,500,966**
8	N/A	**63,434,401,551,198**
9	N/A	**3,362,023,282,213,494**

Table 3, Effect of DOF on Unique and Total Points Reachable

Legend:

Roman = Computed by MATLAB

Bold = Estimated

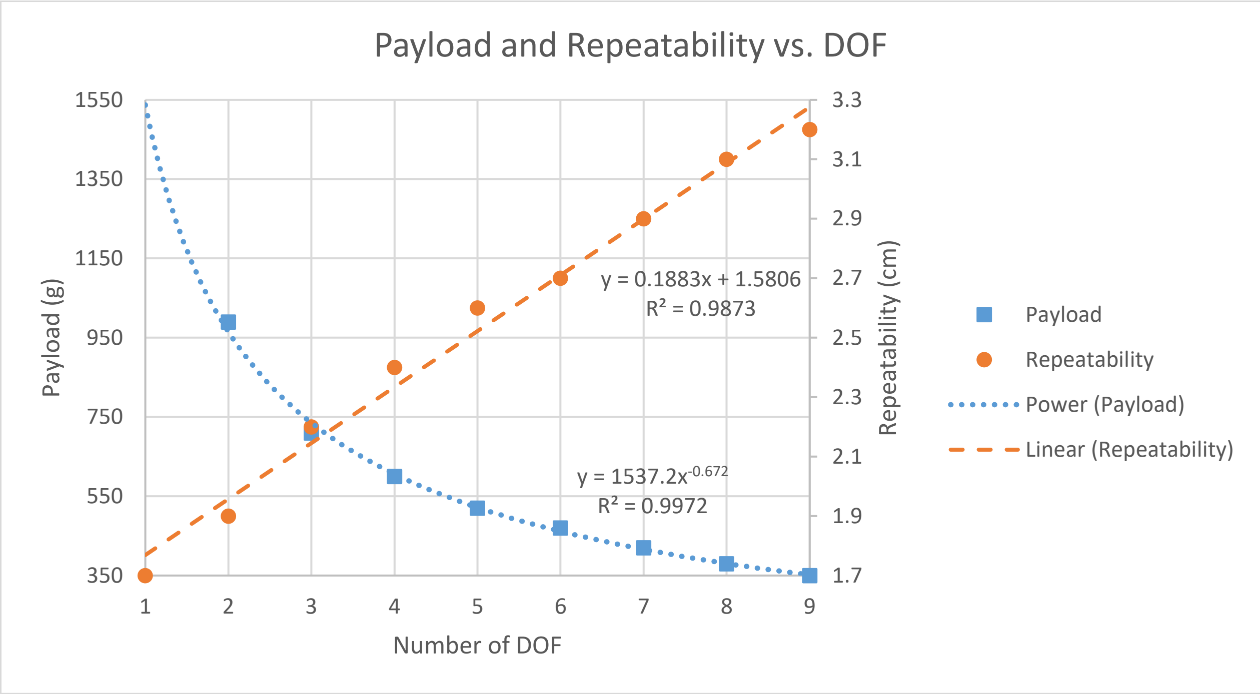
When one examines the total points column, it can be found that each of the computed values are exactly 53 times the previous. This can be explained by the fact that with each new DOF, the number of total iterations should be multiplied by a constant, which is equal to the range of the iterations divided by the iteration increment:

2*(pi/2) / 0.06 ≈ 52.36

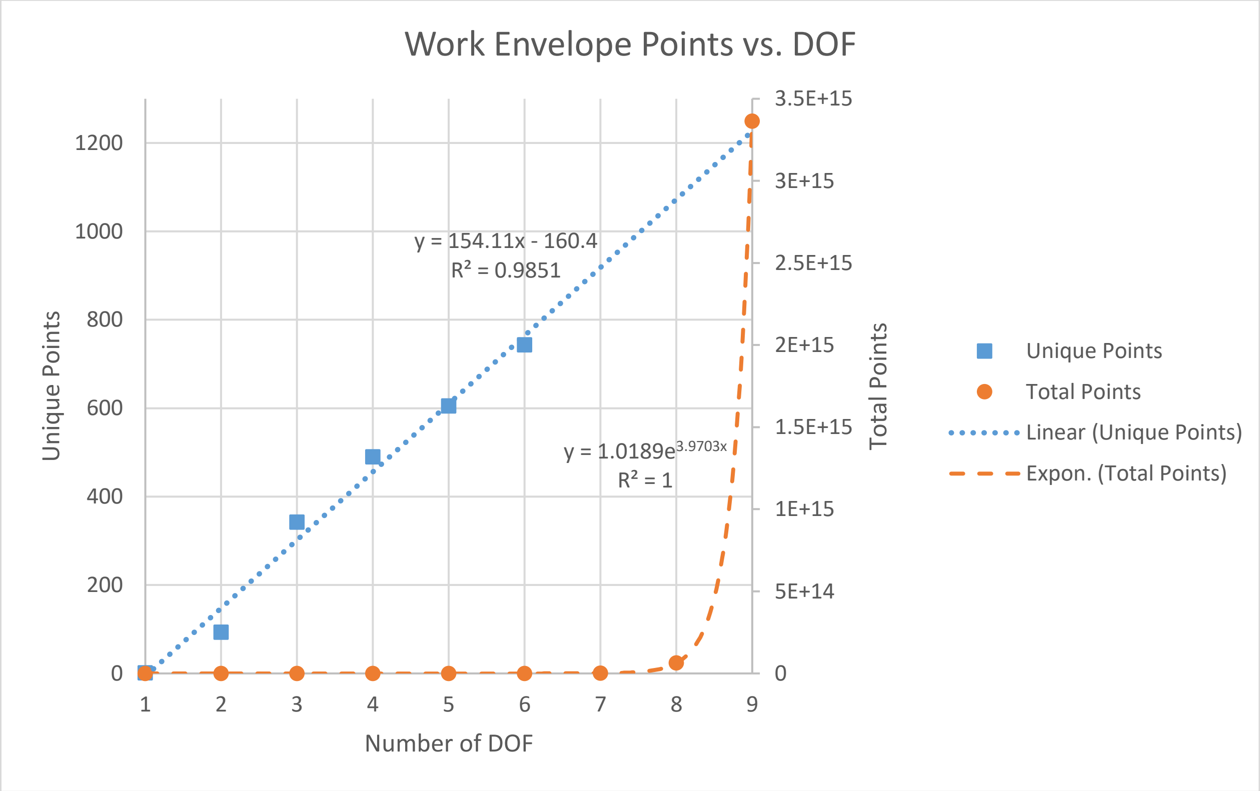
When rounded upwards to the nearest integer, this yields 53. Hence, for the robots with DOF 7 and above, the values were calculated by multiplying the previous by 53, instead of using MATLAB.

Additionally, some of the unique point values haven't been given values, because the MATLAB computation time increases geometrically with each new DOF, meaning that those with a large number take too long to compute practically. However, as shown in the analysis section below, the values form a clear linear trend, which will suffice in extrapolating the greater values.

Analysis



Graph 1, Payload and Repeatability vs. DOF



Graph 2, Work Envelope Points vs. DOF

To be able to compare the different variables and deduce the optimal value for the DOF, they need to have equivalent scales. Each of the variables must carry the same weight in this calculation, because the robot in question is to be as balanced and well-rounded as possible. Hence, each of the variables will be converted to a scale from 0 to 1, by dividing their best fit curve equations by their maximum values (in the DOF domain of 1 to 9).

Additionally, as a value of 1 will be the desirable/preferable value for each variable, the expressions will be subtracted from 1 if they are the opposite. For example, the scaled repeatability expression will be subtracted from 1, because larger repeatability numbers mean greater errors, which are not preferable, and must therefore be expressed by smaller index values.

Payload

The payload has a maximum value of 1537.2 when the DOF is 1. Although in practice, when the DOF is 1, the payload should be infinite, one can still find a value for the payload by extrapolating the best fit curve to x=1.

$B_1 = 1537.2x^{-0.672} / 1537.2 = x^{-0.672}$

Repeatability

The maximum value occurs when the DOF is 9. The repeatability value from the best fit curve at this x value is $y_0 = 0.1883*9 + 1.5806 = 3.2753$

$B_2 = 1 – ((0.1883x + 1.5806) / 3.2753)$

Work Envelope – Unique Points (Reachable Space)

The maximum value happens at x=9, and is y_0 = 154.11*9 – 160.4 = 1226.59

B_3 = (154.11x - 160.4) / 1226.59

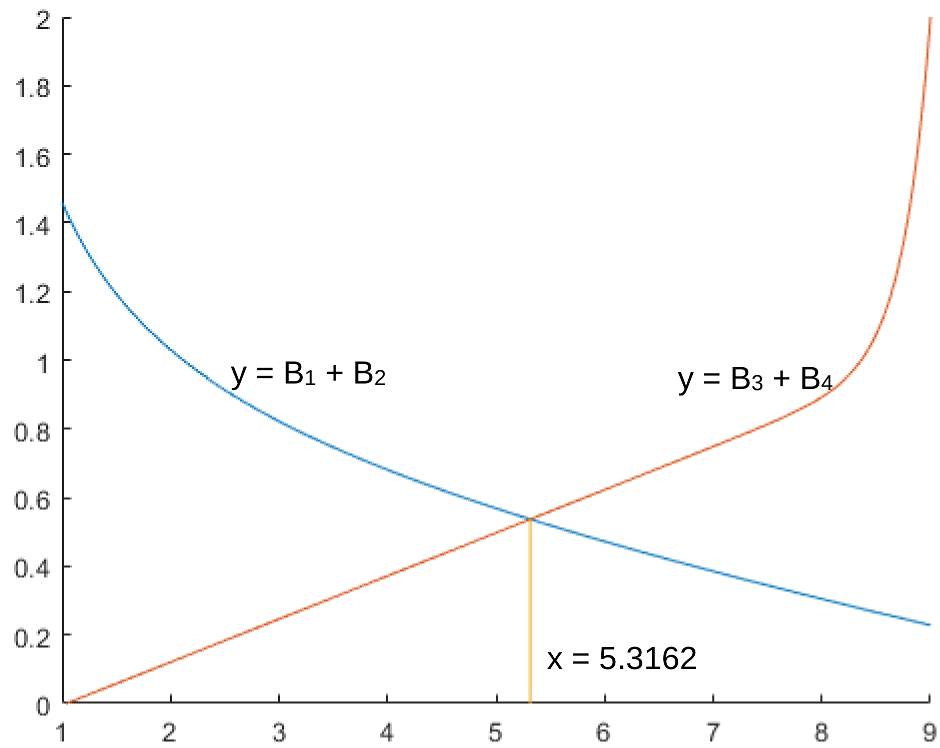
Work Envelope – Total Points (Flexibility)

The maximum value happens at x=9, and is $y_0 = 1.0189e^{3.9703*9} \approx 3.36237\ldots$ E15

$B_4 = 1.0189e^{3.9703x} / y_0 = e^{3.9703(x-9)}$

Unifying Expressions

As both B_1 and B_2 (payload and repeatability) are decreasing functions of the DOF, and B_3 and B_4 (unique and total points) are increasing, both pairs are added individually, and then the intersection of the two sum functions is found.



Graph 3, Graph of $y=B_1+B_2$ and $y=B_3+B_4$

When the intersection of the two functions is computed numerically, this result is found:

$x = 5.3162 \approx 5$

Conclusion

In this extended essay, the relationship between the number of degrees of freedom in an articulated robot arm, and the repeatability, payload, and work envelope has been investigated. This has been done by modelling each of these variables, either through straightforward calculations or through MATLAB computations. These models were converted to index values between 1 and 0, and were then manipulated so that the DOF value which yielded the most balanced robot in terms of the various specifications could be found.

The optimal number of degrees of freedom for the described robot has been found to be 5. This matches what had initially been expected (5 to 6), and what is typically used in the industry. It can therefore be concluded that the most commonly used robots in industries are in fact the most balanced and well-rounded robots, in terms of specifications.

It should be noted that the robot configuration and the specifications of the motors used are bound to have an effect on the results. Therefore, the fact that the robot being modelled was based on hobby-scale robots has the implication that the result may not be completely applicable to the larger industrial robots. The investigation could therefore be improved by studying industrial robots with specific industrial servo motors. Further improvements could be made by removing the assumptions made throughout this essay, such as the assumption that all the link lengths were of equal length and mass.

Works Cited

Kucuk, Serdar and Zafer Bingul. "Robot Kinematics: Forward and Inverse Kinematics." Cubero, Sam. *Industrial Robotics: Theory, Modelling and Control*. Pro Literatur Verlag, 2006. <http://www.intechopen.com/books/industrial_robotics_theory_modelling_and_control/robot_kinematics__forward_and_inverse_kinematics>.

"MG996R Digital Servo Metal Gear." 25 September 2016. *Ebay.* <http://www.ebay.com.au/itm/MG996R-Digital-Servo-Metal-Gear-Steering-Servo-Helicopter-RC-Car-Boat-Model-/191862737980>.

"Robot Arm Configurations." 2016. *Robot Basics.*

RobotWorx. "It's All About That Strength - Robot Payloads." n.d. *RobotWorx.* <https://www.robots.com/articles/viewing/it-s-all-about-that-strength-robot-payloads>.

—. "What are the main types of robots?" n.d. *RobotWorx.* <https://www.robots.com/faq/show/what-are-the-main-types-of-robots>.

—. "What is a work envelope?" n.d. *RobotWorx.* <https://www.robots.com/faq/show/what-is-a-work-envelope>.

—. "What is robot repeatability?" n.d. *RobotWorx - Used Robots.* <https://www.used-robots.com/faq/question/what-is-robot-repeatability>.

Ross, Larry, et al. *Robotics: Theory and Industrial Applications*. Goodheart-Willcox, 2010. <http://www.g-w.com/pdf/sampchap/9781605253213_ch02.pdf>.

United States Department of Labor. "Industrial Robots and Robot System Safety." n.d. *Occupational Safety & Health Administration.* <https://www.osha.gov/dts/osta/otm/otm_iv/otm_iv_4.html>.

Vex Robotics. "10.2: Degrees of Freedom." n.d. *Vex Robotics Curriculum.* <http://curriculum.vexrobotics.com/curriculum/lifting-mechanisms/degrees-of-freedom>.

Wysk, Richard. "Introduction to Robotics." n.d. *PennState College of Engineering.* Powerpoint Document. <www.engr.psu.edu/cim/ie450/ie450pp8.ppt>.

Appendix

MATLAB Source Code for Work Envelope Calculations

```
% ee_work_envelope:
%   Used to calculate the work envelope of an articulated robot arm
%   based on the number of points reachable (unique, and total), with
%   the points rounded to a 3D grid (nearest 5cm) for consistency.
%   It is assumed that the link lengths are all equal (= reach / DOF)
%   DOF:
%     Number of degrees of freedom / joints
%   reach:
%     Total reach of arm, in cm
%   angleIncrement:
%     The value to increment the angles by while
%     iterating. Larger values will result in skipped points on
%     the grid, while smaller values will require more computation time.
%   maxAngle:
%     The maximum angle that a joint can rotate to, from its center
%     position. Applied to all the joints, for consistency.
function ee_work_envelope(DOF, reach, angleIncrement, maxAngle)
    tic; % Start timer (for benchmarking)

    draw_figure();

    coordinates = iterate_angles_first(DOF, angleIncrement, maxAngle, reach/DOF);

    iterate_coordinates(coordinates);
    colormap(jet);
    colorbar;

    view(0,90);

    toc;
end

% draw_figure:
%   Prepares the graphical window
function draw_figure
    global az el;

    figure; % launches default window

    title('3D Diagram of Work Envelope');
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    grid on;

    axis([-50 50 -50 50 -30 60]);

    h = rotate3d;
    h.ActionPostCallback = @rotate_event;
    h.Enable = 'on';

    uicontrol('Style','text','String','Azimuth','Position', [0 15 50 20]);
    uicontrol('Style','text','String','Elevation','Position', [52 15 50 20]);
    az = uicontrol('Style','edit','String','0','Position', ...
        [0 0 50 20],'Callback',@az_type_event);
```

```
    el = uicontrol('Style','edit','String','0','Position', ...
        [51 0 50 20],'Callback',@el_type_event);
end

% iterate_angles_first:
%   A version of the iterate_angles function (see below), which is only to
%   be used with the first DOF.
%   Its loop uses parallel workers, and it also has code for the progress bar.
function coordinates = iterate_angles_first(dof, angInc, maxAng, linkLength)

    % The following is a 3-dimensional matrix representing the coordinates.
    % Add 31 to each of the coordinates (x, y, and z) to get the matrix
    % index (i.e. x=-30 has an index of x=1, and x=30 has x=61, in the matrix). The
value
    % contained in the matrix represents how many instances of that point was
    % calculated (i.e. counting multiple solutions).
    coordinates = zeros(61, 61, 61);

    % First joint only iterates from -maxAng degrees to 0 degrees,
    % to optimize the computation. The values are reflected
    % to the other side later, on line 138
    parfor O_int = 0 : int8(maxAng/angInc),
        O = -double(O_int)*angInc;
        coordinates_partial = zeros(61, 61, 61);

        T = get_transformation_matrix(1, dof, O, linkLength); % Calculate next
transform matrix

        if dof > 1, % If not last joint, keep going
            coordinates_partial = iterate_angles(2, T, coordinates_partial, dof,
angInc, maxAng, linkLength);
        else % If last joint, calculate end-effector position and round to
             % grid (nearest 5cm)
            x = round(T(1,4)/5, 0) * 5 + 31;
            y = round(T(2,4)/5, 0) * 5 + 31;
            z = round(T(3,4)/5, 0) * 5 + 31;
            coordinates_partial(x, y, z) = coordinates_partial(x, y, z) + 1;
        end

        coordinates = coordinates + coordinates_partial;
    end
end

% iterate_angles:
%   Recursive function that iterates the joint angles with angleInc
%   increments, to find all the possible joint combinations and
%   their respective end-effector positions.
%   jointNum:
%     Indicates which joint is being called for; is necessary for
%     the recursive function. 1 = first joint, dof = last joint
%   prev_T:
%     The transformation matrix up until the joint in question.
function coordinates_partial = iterate_angles(jointNum, prev_T, coordinates_partial,
dof, angInc, maxAng, linkLength)
    min = maxAng;
    max = maxAng;

    for O = -min : angInc : max,
        T = prev_T * get_transformation_matrix(jointNum, dof, O, linkLength); %
Calculate next transform matrix

        if jointNum < dof, % If not last joint, keep going
```

```
            coordinates_partial = iterate_angles(jointNum + 1, T, coordinates_partial,
dof, angInc, maxAng, linkLength);
        else % If last joint, calculate end-effector position and round to
             % grid (nearest 5cm)
            x = round(T(1,4)/5, 0) * 5 + 31;
            y = round(T(2,4)/5, 0) * 5 + 31;
            z = round(T(3,4)/5, 0) * 5 + 31;
            coordinates_partial(x, y, z) = coordinates_partial(x, y, z) + 1;
        end
    end
end

function iterate_coordinates(coordinates)
    pointCount = 0;
    points = zeros(4, 500000);
    i = 1;

    coordinates = coordinates + flip(coordinates,2);

    for x = -30 : 5 : 30,
        for y = -30 : 5 : 30,
            for z = -30 : 5 : 30,
                v = coordinates(x+31, y+31, z+31);
                if v > 0,
                    points(1,i) = x;
                    points(2,i) = y;
                    points(3,i) = z;
                    points(4,i) = log(v);
                    i = i + 1;
                    pointCount = pointCount + v;
                end
            end
        end
    end
    fprintf('%i unique, %i total points found\n', i-1, pointCount);

    scatter3(points(1,:), points(2,:), points(3,:), 40, points(4,:), 'o', 'filled');
end



function T = get_transformation_matrix(jointNum, dof, Q, d)
    switch jointNum
        case 1
            T = T_base(Q,d);
        case 2
            T = T_turn_side(Q,d);
        case 3
            if dof <= 6
                T = T_repeat_side(Q,d);
            else
                T = T_turn_up(Q,d);
            end
        case 4
            if dof <= 5
                T = T_repeat_side(Q,d);
            elseif dof == 6
                T = T_turn_up(Q,d);
            else
                T = T_turn_side(Q,d);
            end
        case 5
            if dof == 6
```

```
                T = T_turn_side(Q,d);
            else
                T = T_turn_up(Q,d);
            end
        case 6
            if dof == 6
                T = T_turn_up(Q,d);
            else
                T = T_turn_side(Q,d);
            end
        case 7
            T = T_turn_up(Q,d);
        case 8
            T = T_turn_side(Q,d);
        case 9
            T = T_turn_up(Q,d);
    end
end

function T = T_base(Q,d)
    T = [cos(Q) -sin(Q) 0 0; sin(Q) cos(Q) 0 0; 0 0 1 d; 0 0 0 1;];
end

function T = T_turn_side(Q,d)
    T = [cos(Q) -sin(Q) 0 -d*sin(Q); 0 0 -1 0; sin(Q) cos(Q) 0 d*cos(Q); 0 0 0 1;];
end

function T = T_repeat_side(Q,d)
    T = [cos(Q) -sin(Q) 0 -d*sin(Q); sin(Q) cos(Q) 0 d*cos(Q); 0 0 1 0; 0 0 0 1;];
end

function T = T_turn_up(Q,d)
    T = [cos(Q) -sin(Q) 0 0; 0 0 1 d; -sin(Q) -cos(Q) 0 0; 0 0 0 1;];
end



% The functions below make the UI interactive

function rotate_event(~,evd)
    global az el;
    newView = round(evd.Axes.View);
    az.String = newView(1);
    el.String = newView(2);
end

function az_type_event(source,~)
    [~, el] = view;
    view(str2double(source.String), el);
end

function el_type_event(source,~)
    [az, ~] = view;
    view(az, str2double(source.String));
end
```