New Software to Assess Equations of Motion

J. R. Yujnovich, N.W. Scott & B.J. Stone
The University of Western Australia

Abstract

By far the most common form of computer-based assessment software is multiple-choice. Although this is convenient for "digital" marking, the resulting educational experience for the student can leave something to be desired. Therefore the authors are investigating ways of allowing the student of engineering dynamics to respond to problems in a more "free-form" manner. The example presented in this paper allows students to enter the equations of motion for a particle in contact with a plane. The student must follow certain notation conventions, but is not prompted to use any particular co-ordinate system. From the resulting strings it is possible to infer not only which co-ordinate system the student is implicitly using, but whether it is used consistently, or if some other error has been made. Specific diagnostic feedback is provided in the case of an error. Results from student use of the software as part of an assessed course are presented.

1 Background

During 1995 at the University of Western Australia the conventional tutoring system for first year engineering dynamics was replaced by a computer based assessment system which provided students with diagnostic feedback[1]. This tutorial system has the following features:

1 Students log in using a password.

2 All the computer terminals are in one large room, which encourages student-student interaction.

3 Students attempt problems that are presented on the computer screen. The current problem must be solved before moving to the next.

4 Each student has a unique set of numerical values for every problem.

5 Students enter answers that are always a number with units e.g. “3.2 m/s”.

6 There are typically eight ‘lead-up’ problems in each set, followed by two assessed problems. The assessed problems are marked based on the number of attempts required to obtain the correct answer. Although the ‘lead-up’ questions are not marked, they must be completed before the assessed questions can be attempted. In 1995 the assessed problems count for 20% of the year’s mark.

7 The ‘lead-up’ questions form a carefully chosen sequence that explores each of the pitfalls of the assessed problems. An example question is shown in Figure 1.

8 The software surrounding the ‘lead-up’ problems is often able to ‘diagnose’ the difficulty with an incorrect answer, based on common student errors. If this occurs then the student can immediately view very specific explanatory material related to the misconception.

9 All student actions are recorded by a central ‘server’ computer. Students who fall behind are thus easily spotted. This monitoring feature has allowed students in trouble to be contacted by telephone or letter and arrangements made for personal tuition or other assistance. A unique display maintained by the serving computer also allows teaching staff to immediately determine if any of the problems are causing the class general difficulty.

10 A human tutor is present during assigned tutorial hours, to offer help if the computer’s diagnosis and associated help is insufficient. In practice the tutor is often idle in a class of 40 students.

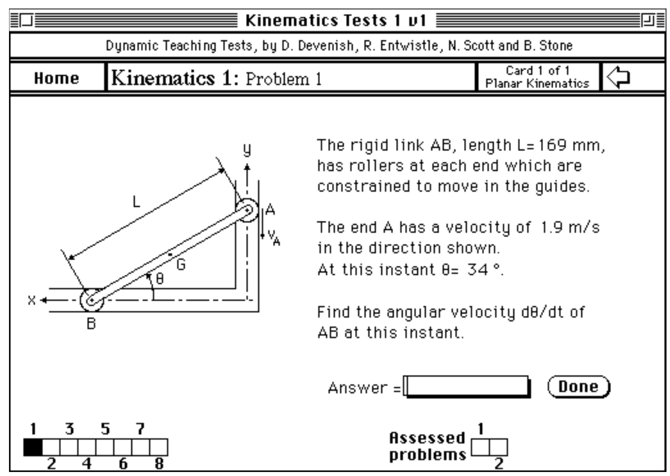

Figure 1 A typical ‘lead-up’ problem card.

The idea behind the diagnostic feedback was to determine a misunderstanding by working back from an incorrect numerical answer. This is best summarised by the diagram shown in Figure 2. The student answer is compared with a number of areas in the set of real numbers. The areas labelled Answer, Error 1 and Error 2, are simply numerical values associated with the actual answer, and two possible errors which the problem will detect. These values have tolerances of plus and minus two percent. If the student answer should fall within one of theses limits then either the correct answer feedback is given, or the feedback associated with the particular error is shown. If the answer does not fall within one of these limits, then it is assumed to be an arithmetic error.

The use of these computer based tutorials was well received by the students. Figure 3 shows the response to one of the questions on a survey conducted by an independent group within the university.

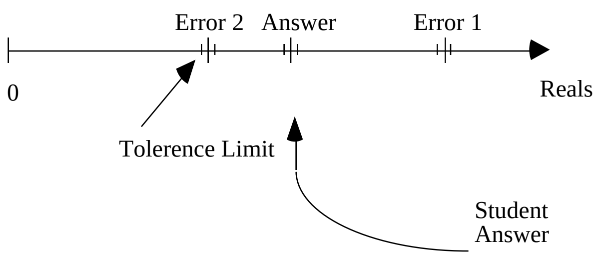

Figure 2 Pictorial representation of numerical based checking.

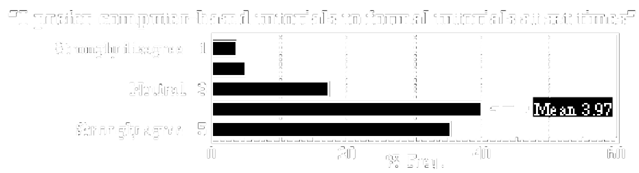

Figure 3 Student survey result.

However the method does have some disadvantages. If a student gets a problem wrong, then unless the package is able to determine which erroneous path the student has followed, no useful feedback is given to the student. The student is no better off for having attempted the problem. Checking for misunderstandings using only the numerical answers is not the most effective way to instil a deep understanding of the topic. There could be much learned from the student being asked to form the solution equations and input them as the answer.

As a result a final year project was undertaken to open up a new avenue of assessment with a view to it being included in the system. The aim was to produce a trial problem involving an equation checker with diagnostic feed back. If the equation checker package proved successful as a teaching tool, then there could be a series of such problems produced, within a single package. This could then enable the students to gain a thorough understanding in a variety of topics which allow for parametric equations as answers.

It was also an aim of the project that the program should be able to store the information when it was being used, for future analysis. By analyzing student responses it should become clear which errors are being successfully detected and which errors, previously not thought of, should be checked.

2 The Equation Checker Program

The Equation Checker package was initially created as a text based Think Pascal program, which was later adapted, as an X function, to fit into the HyperCard stack environment of the computer based assessment set up. A student would be given the problem description and then simply asked to enter in the two solution equations. In the early development stages of the Equation Checker it was unclear as to whether the method for determining the correctness of a set of equations could actually be applied successfully. For this reason the method of checking through a set of equations was only applied to one particular problem, which involved Newton's second law of motion.

During the program development it was decided that a text based system would be adequate for testing purposes, and for this reason the designing of a suitable user interface was left until the latter stages of the project. The actual problem card, along with solution cards and the help card, were produced just prior to the stack being placed onto the server.

The Equation Checker program is based on string based parsing. This in effect means the program may be said to have no mathematical knowledge. The ability of a program to perform using actual mathematical criteria would require much more detailed and complex parsing of the equation strings, in order to recognize mathematical symbols. It may be that with the addition of the power of such packages as Maple and Mathematica, the theory behind the Equation Checker may be very useful. Actually showing that the Equation Checker can perform the required checks successfully is of great benefit to the future of other similar packages.

3 The Equation Checker Problem

The problem used as the basis for the program is shown in Figure 4. This problem was given to the students as a Kinetics revision problem. The student is given as little information as possible and is asked to enter the equations of motion for the problem, using Newton's second law. There are a number of ways in which the problem can be approached.

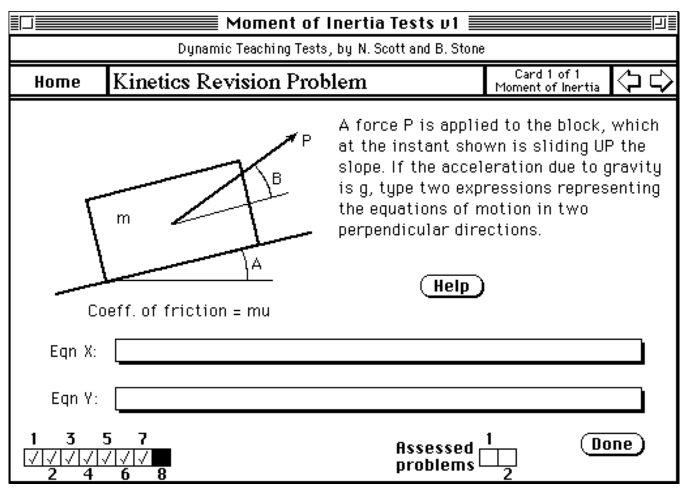

Figure 4 The Equation Checker Problem

It was hoped that the Equation Checker would be able to check for the majority of the misunderstandings which were likely to be made. The success of this package depended upon the ability to firstly detect the error and secondly to give clear and concise feedback to the student. The intention was that the feedback should allow the student to correct any error and so be able to continuously improve their solution.

Use of the ‘help’ button on the problem card of Figure 4 results in the card shown in Figure 5 being displayed.

As shown in Figure 4, the students were given the variable names of the mass, applied load, friction coefficient, and gravitational constant, as well as the angles. Using this information the students would have to choose an axes system and derive the parametric equations of motion for the problem according to the axes system chosen. The equations are entered into the two boxes 'Eqn X' and 'Eqn Y', shown in Figure 4.

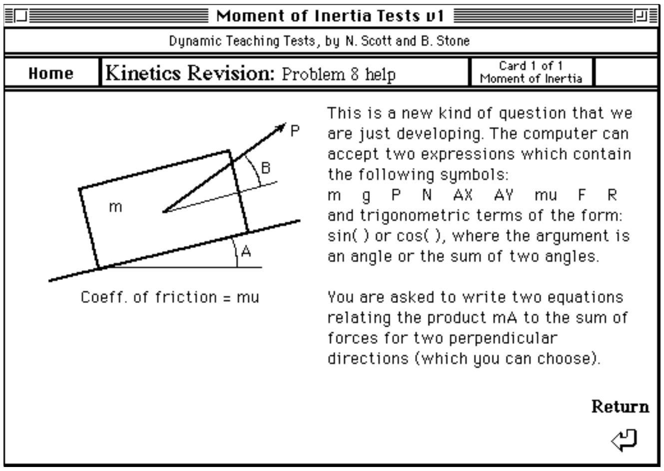

Figure 5 Equation Checker - The help card.

Any package which allows a set of parametric equations to be entered should allow the user to have freedom to choose their own variables. However, there were some limitations on the variables which could be used. This was not to limit the user's choices but to aid significantly in the string based checking procedures. The help card, shown in Figure 5, clearly defines the variables the software could recognize. This help card does not, however, give details of which variables are to be used and which are to be overlooked. The reason for such a detailed help card was that the problem was completely different to anything the students had encountered previously.

Note that students were not prompted to choose a particular X-Y coordinate system to work with. One of the aims of this package was to be able to check the student’s equations for correctness regardless of the axis system chosen.

4 Possible Misunderstandings

By anticipating the thought processes a student may have it is possible to determine which components of the equations of motion are likely to be entered incorrectly and if so what forms these mistakes may take. Since each equation of motion has several terms, and since several axis systems may be used (both correctly and incorrectly), a great number of ‘solutions’ must be considered.

The method used to produce the checking algorithms was simply exhaustive string based analysis of the solution set of equations for each axes system which was to be checked. For each term in the equations of motion a procedure was developed to perform the necessary string based checking. The program did not check for mathematical symbols, but checked for strings within the terms of the equations. This is an area which could and should be improved.

The structure of the Equation Checker program is shown in Figure 6. Each procedure determines the correctness of a particular component of the solution equations. The Check Force Term procedure makes a judgment about which set of axes has been used and this fact is used by the other procedures.

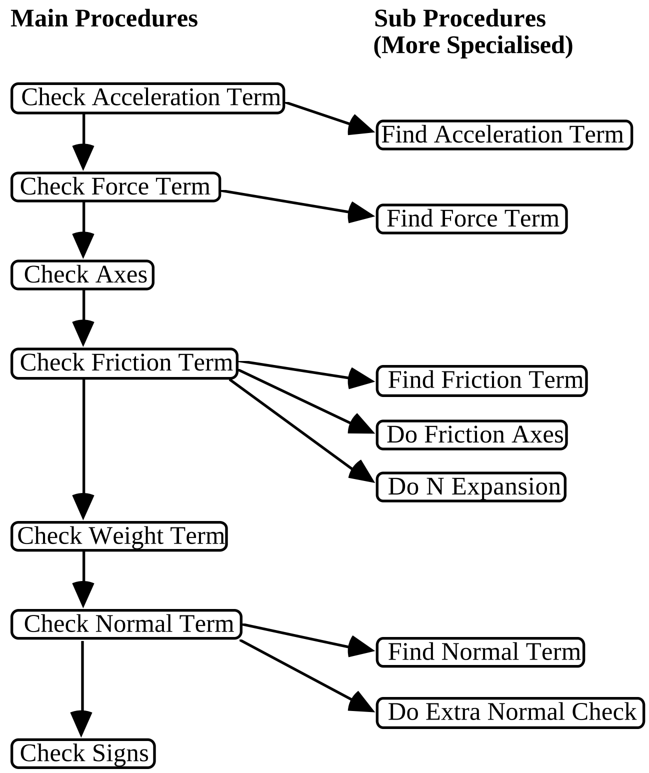

Figure 6 The Equation Checker Program Structure

5 Results

Examples of some student sessions are given in Appendix A. These show the degree of interaction and diagnostic help. For one of the two students a prolonged session occurred before the correct solution was achieved.

A number of students were interviewed informally to get some feedback on the Equation Checker, once the deadline for the stack containing the problem had passed. Feedback from the students on the Equation Checker proved, in general, to be very positive.

Most students found the Equation Checker to be an effective revision tool. A number of students also expressed the point that the new form of problem was refreshing. Moving away from the numerical-answer problems was a good way to break up the continuous stream of standard problems.

A number of students also said that once the free body diagram was drawn the problem became very easy. One student said:

> 'I firstly tried to hack through the problem. I then realized a free body diagram is needed to get the answer.'

This is very significant as the student has discovered that a problem is more easily solved when a free body diagram is used.

In general the students who received feedback mentioned that it clearly stated where the error lay – which is very encouraging. The students were able to correct their equations from the feedback given. One student said:

'The feedback was clear and each point was picked up'

Some students found they had trouble entering the correct form of the friction term. There may be a need for there to be a clearer explanation of the format of the friction term. However this is moving away from the main aim of the project, which was to allow the student the freedom to define their own variables and also choose an axes system.

Some students found it annoying trying to get the equation in the correct format, for the program to check. This is one aspect of the Equation Checker which could be dramatically improved, quite easily. One particular student felt harshly done by. She entered equations and was told of a particular error. However, the feedback was not specific enough and the student did not change it, but instead tried to put different forms of the same terms in. This is a major drawback of the program at present. When she started afresh and re-entered the equations completely she was able to get the correct answer, which *looked* very close to the initial input. Consequently she felt that she had entered the same equations both times and the program just accepted the second equations over the first, for no reason. The problem lay in the feedback given.

Overall, the Equation Checker problem was very well received by the students. The new form of problem was welcomed as an effective teaching tool. The Equation Checker proved to be successful in checking a set of parametric solution equations.

A more detailed analysis of student answers and errors is given in the thesis by Yujnovich. It is clear from these results that the approach of using equation checkers within dynamics problems has many benefits. Not least is the information given to lecturers on what their students are doing and misunderstanding.

7 References

1 Devenish DG, Entwistle RD, Scott N & Stone BJ (1995), 'An assessment package with diagnostic facilities', *Proceedings of International Conference on Computers in Education*, Selected Applications Track Papers, Singapore, December 1995, pp 101-107.

2 Yujnovich J. R., '*Design of Computer Based Teaching Packages for Engineering Dynamics 100*'. Final year honours thesis, Department of Mechanical and Materials Engineering, The University of Western Australia, 1995.

8 Appendix A: Some typical student sessions

The serving computer recorded all student entries and responses made by both the diagnostic software and the student. These records are reasonably self explanatory and three examples are included here. The conclusions drawn by the software are also shown. The last of these shows the extended interaction session of one student before the correct solution was achieved. The asterisk indicates which of the dialogue options were chosen.

Student A

Friday, 20 October 1995
10:57:32 MSG Pcos(A+B)-Nsin(A)-Fcos(A)=mAX
MSG -mg+Ncos(A)-Fsin(A)+Psin(A+B)=mAY
MSG Equation X consists of the following terms:
10:57:36 MSG Term 4 is -mAX
MSG Equation Y consists of the following terms:
MSG Term 5 is -mAY
MSG Axis is of type 1 and both friction terms have been found with F used.
10:57:37 MSG Friction in X: Fcos(A)
MSG Friction in Y: Fsin(A)
MSG Found both the normal force terms.
MSG All checking of signs is done by moving the terms to the same side of the equation as the acceleration term.
MSG The applied force term and the friction force term are in the opposite direction.
MSG No errors were detected. At the end of the main procedure.
10:57:38 Attempt at problem 16080; DN = 1; SN = 1; ED; LD; live = 33:42
dialog: Excellent! (Solution, Same, *Next)

Student B

Tuesday, 17 October 1995
10:34:34 stack "Moment of Inertia Tests v1"; card "16080_JasonHelp"; ; openCard
10:34:57 openCard
10:36:25 mAY=Psin(B)-mgcos(A)
MSG mAX=Pcos(B)-muN-mgsin(A)
MSG Equation X consists of the following terms:
MSG Term 1 is +mAY
MSG Term 2 is -Psin(B)
10:36:26 MSG Term 3 is +mgcos(A)
MSG Equation Y consists the following terms:
MSG Term 1 is +mAX
MSG Term 2 is -Pcos(B)
MSG Term 3 is +muN
MSG Term 4 is +mgsin(A)
MSG It is unclear as to which is the X equation,looking at the acclereration terms.
MSG It is unclear as to which is the Y equation, looking at the accleration terms.
MSG Axis is 2 and the friction term, was in the Y equation.
MSG Friction in Y: muN
MSG There is mgcos(A) in X and mgsin(A) in Y, which is correct so far, for the weight term.
J1 There was no normal force terms in either equation.
10:36:41 dialog: There was no normal force terms in either equation. (*Help, OK,)
10:36:42 stack "Moment of Inertia Tests v1"; card "16080_JasonHelp"; ; openCard
10:36:45 openCard
10:37:05 closeStack
10:37:09 CLOS
12:45:18 LOGI
SESA SESN SECP SBDN SLME
12:45:19 SEWS
12:45:20 RESQ
macName: B29
stack "Login"; card "loginCard"; goToProblemAfter 16070
12:45:21 GTPA
12:45:22 openCard
12:45:23 stack "Moment of Inertia Tests v1"; card id 21026; 16000; openCard
12:45:24 openCard
12:48:34 MSG mAX=-mgcos(A)+Psin(B)-mu(Pcos(B)-mgsin(A))
MSG mAY=-mgsin(A)+Pcos(B)
MSG Equation X consists of the following terms:
MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
MSG Term 3 is -Psin(B)
MSG Term 4 is +mu(Pcos(B)
MSG Term 5 is +mgsin(A))
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is +mgsin(A)
12:48:35 MSG Term 3 is -Pcos(B)
G22 There should be two friction terms in the X equation since the normal force term is made up of two components, a weight term and an applied force term.
12:48:51 dialog: There should be two friction terms in the X equation since the normal force term is made up of two components, a weight term and an applied force term. (Help, *OK,)
12:49:11 stack "Moment of Inertia Tests v1"; card "16080_JasonHelp"; ; openCard
12:49:16 openCard
12:49:23 MSG mAX=-mgcos(A)+Psin(B)-muPcos(B)+mumgsin(A))
MSG mAY=-mgsin(A)+Pcos(B)
MSG Equation X consists of the following terms:
12:49:24 MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
MSG Term 3 is -Psin(B)
MSG Term 4 is +muPcos(B)
MSG Term 5 is -mumgsin(A))
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is +mgsin(A)
MSG Term 3 is -Pcos(B)
MSG There is mgcos(A) in X and mgsin(A) in Y, which is correct so far, for the weight term.
J1 There was no normal force terms in either equation.
dialog: You seem to be having trouble with this problem. Do you want to see the solution? (No, *Yes,)

12:49:25 stack "Moment of Inertia Tests v1"; card "16080_solution"; ; openCard
12:49:29 stack "Moment of Inertia Tests v1"; card id 13624; ; openCard
12:49:44 stack "Moment of Inertia Tests v1"; card id 16560; ; openCard
12:49:47 stack "Moment of Inertia Tests v1"; card id 14769; ; openCard
12:49:50 stack "Moment of Inertia Tests v1"; card id 16983; ; openCard
12:51:09 openCard
12:51:52 MSG mAX=-mgcos(A)+Psin(B)-F
MSG mAY=-mgsin(A)+Pcos(B)+N
MSG Equation X consists of the following terms:
MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
12:51:53 MSG Term 3 is -Psin(B)
MSG Term 4 is +F
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is +mgsin(A)
MSG Term 3 is -Pcos(B)
MSG Term 4 is -N
E4 The friction term should be in the equation containing the component of the applied force along the plane not the component vertical to the plane.
12:52:00 dialog: You seem to be having trouble with this problem. Do you want to see the solution? (*No, Yes,)
12:52:10 dialog: The friction term should be in the equation containing the component of the applied force along the plane not the component vertical to the plane. (Help, *OK,)
12:52:34 MSG mAX=-mgcos(A)+Psin(B)-muN
MSG mAY=-mgsin(A)+Pcos(B)+N
12:52:35 MSG Equation X consists of the following terms:
MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
MSG Term 3 is -Psin(B)
MSG Term 4 is +muN
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is +mgsin(A)
MSG Term 3 is -Pcos(B)
MSG Term 4 is -N
E4 The friction term should be in the equation containing the component of the applied force along the plane not the component vertical to the plane.
12:53:31 dialog: The friction term should be in the equation containing the component of the applied force along the plane not the component vertical to the plane. (Help, *OK,)
12:55:10 MSG mAX=-mgcos(A)+Psin(B)-mumAY+mumgsin(A)-muPcos(B)
MSG mAY=-mgsin(A)+Pcos(B)+N
MSG Equation X consists of the following terms:
12:55:11 MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
MSG Term 3 is -Psin(B)
MSG Term 4 is +mumAY
MSG Term 5 is -mumgsin(A)
MSG Term 6 is +muPcos(B)
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is +mgsin(A)
MSG Term 3 is -Pcos(B)
MSG Term 4 is -N
G19 There should be an applied force component in one of ther two friction terms in the X equation. The normal force term is made up of two components.
12:55:32 dialog: There should be an applied force component in one of ther two friction terms in the X equation. The normal force term is made up of two components. (Help, *OK,)
12:55:58 MSG mAX=-mgcos(A)+Psin(B)-F
MSG mAY=-mgsin(A)+Pcos(B)+N
MSG Equation X consists of the following terms:
12:56:04 MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
MSG Term 3 is -Psin(B)
MSG Term 4 is +F
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
12:56:05 MSG Term 2 is +mgsin(A)
MSG Term 3 is -Pcos(B)
MSG Term 4 is -N
E4 The friction term should be in the equation containing the component of the applied force along the plane not the component vertical to the plane.
dialog: You seem to be having trouble with this problem. Do you want to see the solution? (*No, Yes,)
12:56:15 dialog: The friction term should be in the equation containing the component of the applied force along the plane not the component vertical to the plane. (Help, *OK,)
1:00:12 MSG mAX=-mgcos(A)+Psin(B)-muPcos(B)+mumgsin(A)
1:00:13 MSG mAY=-mgsin(A)+Pcos(B)+N
MSG Equation X consists of the following terms:
MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
MSG Term 3 is -Psin(B)
MSG Term 4 is +muPcos(B)
MSG Term 5 is -mumgsin(A)
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is +mgsin(A)
MSG Term 3 is -Pcos(B)
MSG Term 4 is -N
MSG There is mgcos(A) in X and mgsin(A) in Y, which is correct so far, for the weight term.
K20 The normal force should not be in the same equation as the component of the applied force parallel the the plane.
1:00:26 dialog: The normal force should not be in the same equation as the component of the applied force parallel the the plane. (Help, *OK,)
1:01:18 MSG mAX=-mgcos(A)+Psin(B)+muPcos(B)-mumgsin(A)
1:01:19 MSG mAY=-mgsin(A)+Pcos(B)+N
MSG Equation X consists of the following terms:
MSG Term 1 is +mAX
MSG Term 2 is +mgcos(A)
MSG Term 3 is -Psin(B)
MSG Term 4 is -muPcos(B)
MSG Term 5 is +mumgsin(A)
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is +mgsin(A)
MSG Term 3 is -Pcos(B)
MSG Term 4 is -N
MSG There is mgcos(A) in X and mgsin(A) in Y, which is correct so far, for the weight term.
K20 The normal force should not be in the same equation as the component of the applied force parallel the the plane.
1:01:29 dialog: The normal force should not be in the same equation as the component of the applied force parallel the the plane. (Help, *OK,)
1:02:26 closeStack
1:02:30 CLOS
2:10:24 LOGI SESA SESN SECP SBDN SLME
2:10:25 SEWS RESQ
macName: D14
2:10:30 stack "Login"; card "loginCard"; goToProblemAfter 16070
GTPA
2:10:32 openCard
stack "Moment of Inertia Tests v1"; card id 21026; 16000; openCard
2:10:35 openCard
2:13:31 stack "Moment of Inertia Tests v1"; card "16080_JasonHelp"; ; openCard
2:13:34 openCard
2:14:20 MSG mAX=Pcos(B)-Nmu-mgsin(A)
MSG mAY=Psin(B)+N-mgcos(A)
MSG Equation X consists of the following terms:
MSG Term 1 is +mAX
MSG Term 2 is -Pcos(B)
MSG Term 3 is +Nmu
MSG Term 4 is +mgsin(A)
MSG Equation Y consists the following terms:
MSG Term 1 is +mAY
MSG Term 2 is -Psin(B)
MSG Term 3 is -N
2:14:21 MSG Term 4 is +mgcos(A)
MSG Axis is of type 2 and the friction term, was in the X equation, not in the Y.
MSG Friction in X: Nmu
MSG There is mgsin(A) in X and mgcos(A) in Y, which is correct so far, for the weight term
MSG Found Normal term in Y not X.
MSG All checking of signs is done by moving the terms to the same side of the equation as the acceleration term.
MSG Axis is of type 2.
MSG No errors were detected. At the end of the main procedure.
Attempt at problem 16080; DN = 1; SN = 9; ED; LD; live = 48:49
2:14:55 dialog: Excellent! (*Solution, Same, Next)
stack "Moment of Inertia Tests v1"; card "16080_solution"; ; openCard

J.R. Yujnovich

Jason Yujnovich received a first-class honours degree and the Julian Hunka Prize in Mechanical Engineering at the University of Western Australia in 1995. His final year honours thesis was the subject of this paper. He commenced work as an engineer with Western Mining in 1996.

N.W. Scott

Nathan Scott is a postgraduate student and research officer in the Department of Mechanical and Materials Engineering, the University of Western Australia. He is employed full-time writing software to illuminate engineering dynamics through presentation of moving examples. This work is funded by the Australian Federal Government through its national priority reserve scheme.

B.J. Stone

Professor Brian Stone has held the Chair in Mechanical Engineering at the University of Western Australia since 1981. He obtained his doctorate from the University of Bristol in 1968. He has been writing teaching software since 1987, some of which is now used at universities throughout the world. His research interests include vibration suppression and computer simulation of dynamic systems.