Ergonomic Nutcracker

MEC 525 Project

Fall 2015

By

Shashank Sharma

110533045

Table of Contents

Introduction

Cracking of a nut shell is very hard to do when done manually. Humans have used tools to get the job done since ancient times. This project intends to design an ergonomic nutcracker which will produce crack in the nut shell with minimum user's hand force and minimum muscle fatigue.

Product Description

A device used to manually and safely crack open hard outer shell of nuts (walnut, almond, hazelnut, etc.) to access the inner seed without causing injury or stress to the user.

Product Design Specifications

- Weight <= 1kg
  - (IOSH guidelines for one hand static load)
- Break nuts in the range [1in,5in]
  - Usual distribution of nut sizes
- Force exerted by user<= 10kg/cm2
  - Prevent bone, joint, muscle, ligament or tendon related injury
- Torque applied by user<= 13 Nm
  - Maximum torque limit by NASA

Product Design Criteria

- Minimize time taken to process each nut (Weight= 1)
- Cost (Weight= 1)

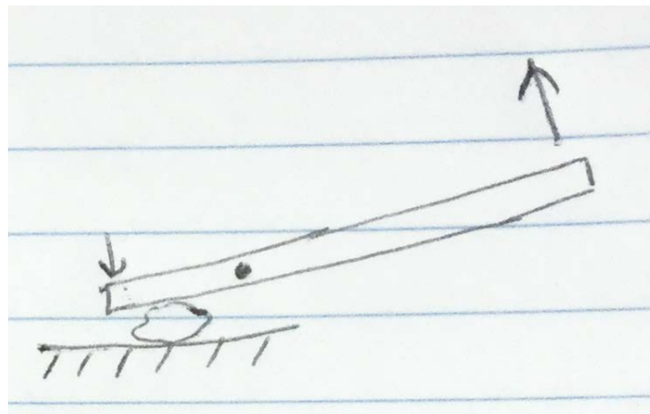
How the product Functions

- Nut Positioning (constraining)
    - Hollow recess
    - Rough surface
- Force application
    - Finger, Palm, Wrist, Legs
- Means of opening nut
    - Cut- Saw, Knife edge (Moving blade/Nut- Inversion)
    - Crack- Levers, Mechanisms, Screw
    - Pull apart- Pierce shell
    - Impact- Potential energy (Spring), Kinetic energy (Pendulum)
- Stray nut shell piece protection
    - Mechanical obstruction

Concept Generation

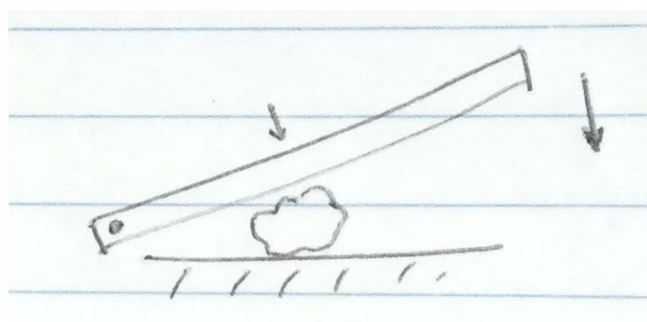
Concept 1

- Using Levers (Class 1)
- Force applied by Hand or Feet
- Recess and rough surface is added to lever to better constraint the nut.



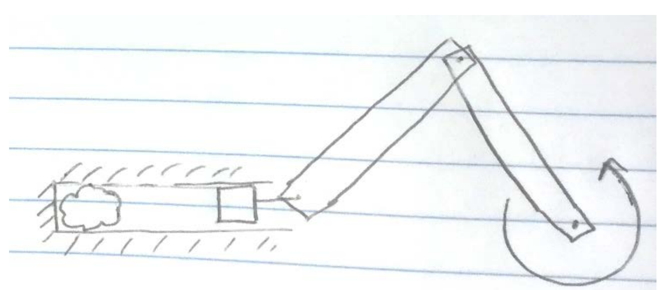
Concept 2

- Using Levers (Class 2)
- Force applied by Hand or Feet
- Recess and rough surface is added to lever to better constraint the nut



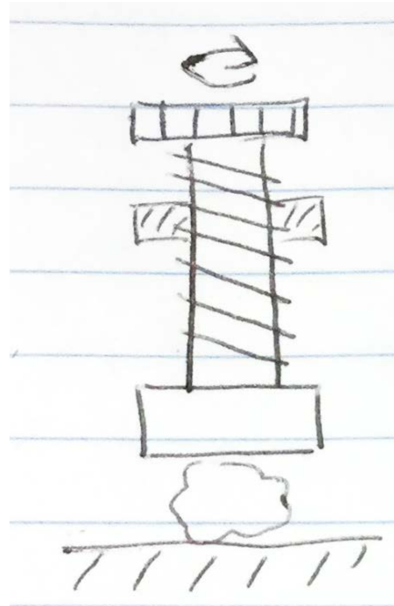
Concept 3

- Using Slider Crank Mechanism
- A gearbox can be added to improve Mechanical Advantage
- Force applied by Palm and Wrist



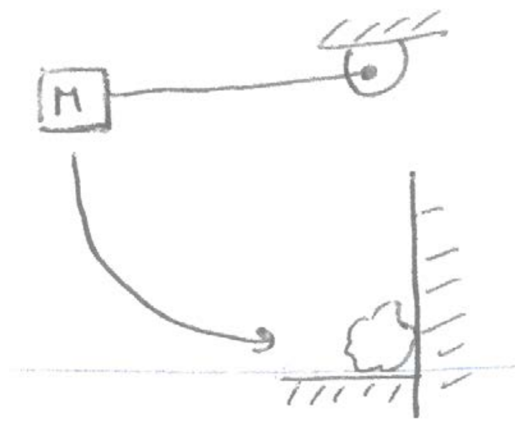
Concept 4

- Using Screw motion
    - Increasing the area at bottom enables us to crack multiple nuts simultaneously
- Torque applied by Fingers
- A recess is added to the surface to constrain the nut.



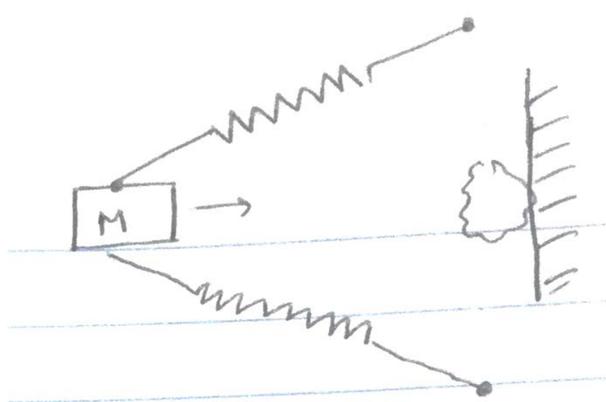
Concept 5

- Using Impact Loading
    - Kinetic Energy
- Force applied by hand in lifting the mass M.
- A recess and rough surface is added to the wall to better constrain the nut



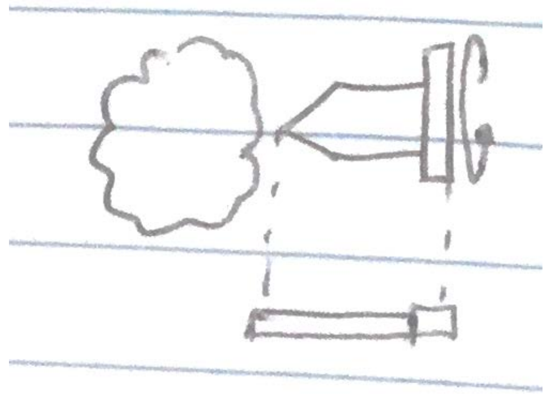
Concept 6

- Using Impact Loading
    - Spring Energy
- Force applied by hand to pull the mass M
- A recess and rough surface is added to the wall to better constrain the nut



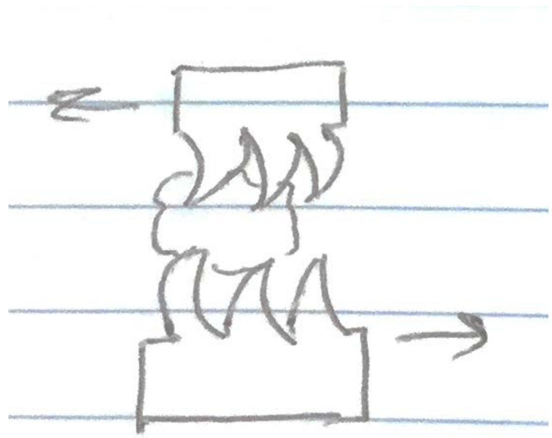
Concept 7

- Using pierce and turn mechanism
- Force applied by hand to pierce the shell and rotate the key like knife



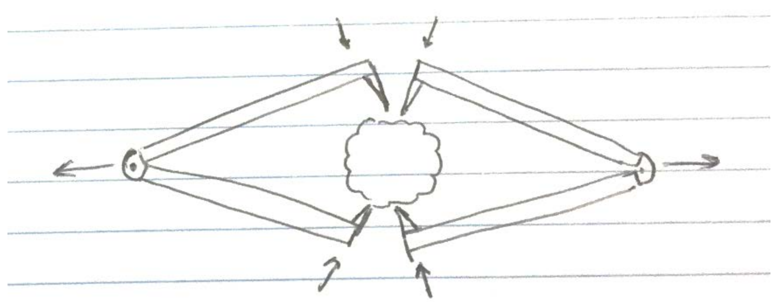
Concept 8

- Using pierce and pull mechanism
- Removes the shell after cracking
- Force applied by hand to pull the blades.
- A cam mechanism can also be used to move the blades by moving a crank



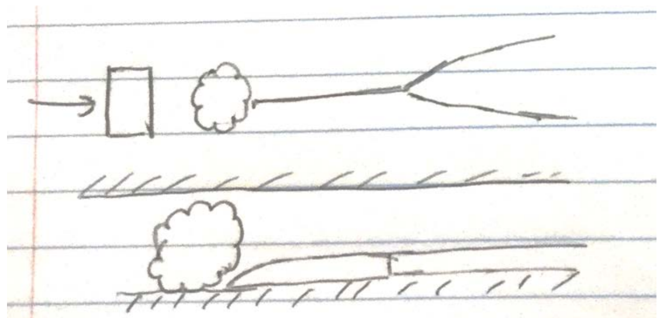
Concept 9

- Using pierce and pull mechanism
- Removes the shell after cracking
- Force applied by Palm to press blades and hand to pull the shell apart



Concept 10

- Pushing the nut over a blade
- Inversion- Instead of the blade moving, the nut moves over the blade



Feasibility Study

S= Specification

C= Concept

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
S1	9	9	5	6	7	7	9	5	7	3
S2	9	9	6	8	7	7	9	7	8	8
S3	7	7	9	9	7	6	6	6	5	8
S4	10	10	6	8	9	9	6	9	5	6
Total	35	35	26	31	30	29	30	27	25	25

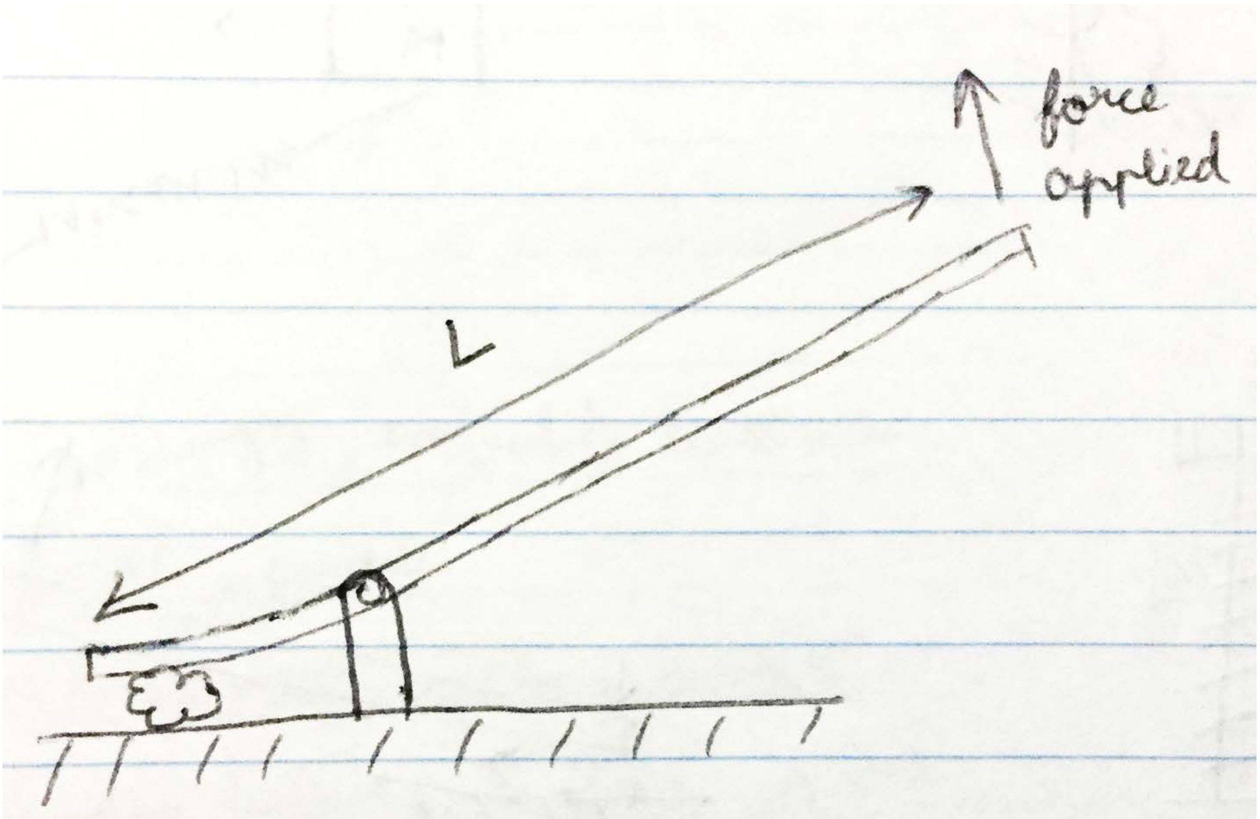
Using Feasibility Study, we narrow down upon Concept 1, 2 and 4 as the most feasible concepts satisfying the product design specifications.

Criteria Evaluation

	Weight	Concept 1	Concept 2	Concept 4
Criteria 1- Time	1	5	4	1
Criteria 2- Money	1	5	5	4
Total		10	9	5

According to Criteria Evaluation, The best possible concept according to the criteria selected is Concept 1.

Detail Design



L= Length of Lever

A= Cross sectional Area of Lever

Cost Function

Cost is proportional to the Volume of Material used to manufacture the product

$$Cost = C \times A \times L$$

Here C= .008 Dollar/cm^2

Time Function

Time taken to process a single nut is inversely proportional to the length as the Larger the length, the more force it applies and the less time it takes to break a nut.

$$Time = K/L$$

Here K=30 s/cm

Constraints

According to single hand usability guidelines,

$$1cm^2 < A < 9cm^2$$

$$10cm < L < 30cm$$

Optimization

$$Minimize\ f(x) = .008 \times A \times L + 30/L$$

$$g1(x) = A - 1$$

$$g2(x) = 9 - A$$

$$g3(x) = L - 10$$

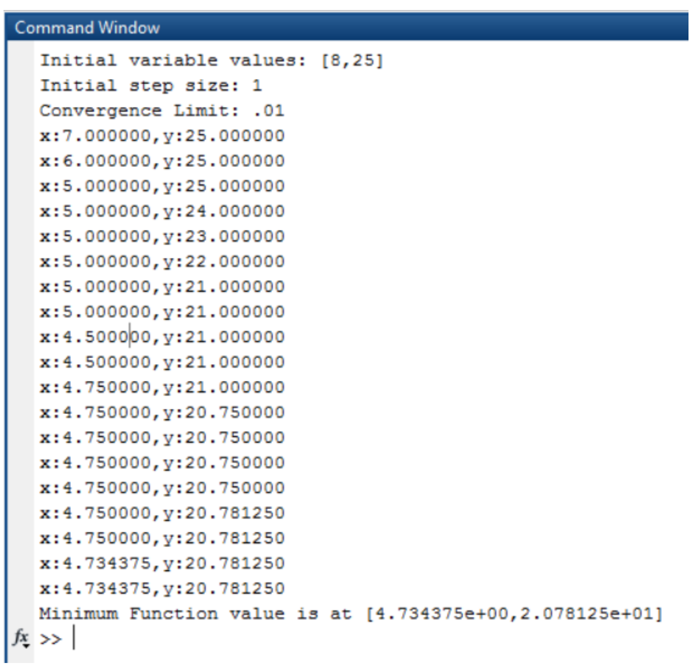
$$g4(x) = 9 - A$$

Taking Initial variable values: [8, 25], Initial step size: 1 and Convergence Limit: .01

Matlab code has been included in the Appendix. Inverse penalty functions have been used to include inequality constraints in the Minimization function.

Result

We find that the objective function is minimum at A=4.73cm^2 and L=20.78cm.



```
Command Window
Initial variable values: [8,25]
Initial step size: 1
Convergence Limit: .01
x:7.000000,y:25.000000
x:6.000000,y:25.000000
x:5.000000,y:25.000000
x:5.000000,y:24.000000
x:5.000000,y:23.000000
x:5.000000,y:22.000000
x:5.000000,y:21.000000
x:5.000000,y:21.000000
x:4.500000,y:21.000000
x:4.500000,y:21.000000
x:4.750000,y:21.000000
x:4.750000,y:20.750000
x:4.750000,y:20.750000
x:4.750000,y:20.750000
x:4.750000,y:20.750000
x:4.750000,y:20.781250
x:4.750000,y:20.781250
x:4.734375,y:20.781250
x:4.734375,y:20.781250
Minimum Function value is at [4.734375e+00,2.078125e+01]
>>
```

Conclusion

By using the basic direct search method involving Inverse penalty function approach, we minimize the objective function. As is seen, the optimal Length=20.78cm and optimal Area=4.73cm^2 .

In this project all the design steps were followed as demonstrated in the class in order to model and optimize the given product design problem.

Appendix

```
function main
clc
clear all
close all

init=input('Initial variable values: ');
step=input('Initial step size: ');
limit=input('Convergence Limit: ');
[x,y]=DirectSearch(init, step, limit);
fprintf('Minimum Function value is at [%d,%d]\n',x,y);
end

function [ x,y ] = DirectSearch( init, step, limit )
x=init(1);
y=init(2);
while (step>limit)
    func=f(x,y);
    fl=f(x-step,y);
    fr=f(x+step,y);
    fd=f(x,y-step);
    fu=f(x,y+step);
    if func>fl
        x=x-step;
    elseif func>fr
        x=x+step;
    elseif func>fd
        y=y-step;
    elseif func>fu
        y=y+step;
    else
        step=step/2;
    end
    fprintf('x:%f,y:%f\n',x,y);
end
end

function [fx] = f(x1,x2)
gx1= x1-1;
gx2= 9-x1;
gx3= x2-10;
gx4= 30-x2;
fx= .008*x1*x2+30/x2+10/gx1+10/gx2+10/gx3+10/gx4;
end
```