

Modeling and simulation of a compliant wall combustion device

Kiran O.V. , Don Daniel*

College of Engineering Trivandrum, India

(Received March 30 2007, Accepted September 8 2007)

Abstract. A mathematical model of the compliant wall combustion device has been constructed & this model simulates the working of the engine so as to plot the various characteristics curves related to this engine by conducting a load test. The iterative algorithm is run real time using a C++ integrated development environment.

Keywords: mathematical model, compliant wall combustion device, iterative algorithm

1 Introduction

The compliant wall combustion device is a miniature internal combustion engine working on a normal Otto cycle. Like ordinary IC engines, there is a cylinder head but the rigid cylinder wall is replaced by an elastic one. The combustion takes place inside this elastic chamber, causing the outward motion of the piston attached to the base of the elastic cylinder, against the elastic forces. The motion is constrained in the axial direction with the help of a helical spring wound over the curved surface of the cylinder. Helical spring also retains the cylindrical shape. Thus it eliminates leakage of fuel through the piston-cylinder gap and mechanical friction from piston-cylinder sliding contact. IDC is the stress free position & ODC is the maximum stress position. The elastic forces oppose expansion but aids compression unlike friction which opposes motion in both directions. More details of the engine can be obtained by referring to US patent application 20060000214.

A mathematical model of this innovative technology has been constructed & this model simulates the working of the engine so as to plot the various characteristics curves related to this technology by conducting a load test. The initial values of various parameters of the engine are set in accordance with the Nemett 15cc single cylinder IC engine. A set of differential equations involving the engine variables and time are derived and instantaneous values of the parameters are used to predict the values in the next instant, assuming a small finite time step. The iterative algorithm is run real time using a C++ integrated development environment.

2 Idealization

The conduction of heat out through the elastic walls is steady. Thermal conductivity doesn't change with temperature. Heat loss through the cylinder head and the cylinder base, which amounts to lesser surface area than the curved portion, is neglected. The combustion takes place instantaneously at constant volume. The mass of the elastic material is neglected. The density of the elastic material is assumed to be a constant, i.e. the volume of the material doesn't change. The shape of the elastic chamber is perfectly cylindrical and is retained always. The expansion of the elastic chamber follows Hooke's law. The elastic cylinder experiences tension only. The Elastic modulus is independent of temperature. The gases are assumed to follow ideal behavior. The properties of the gases in the cylinder are uniform throughout at any instant. Changes in pressure and

* Corresponding author. E-mail address: donstephen@gmail.com.

temperature due to throttling when gases enter or leave through the nozzles are neglected. The pressure drops to atmospheric, instantly during suction and exhaust where the chamber is open to atmosphere. The effect of the gases left after exhaust stroke is neglected. The complications in the structure of the piston, combustion chamber, cylinder head, crank, connecting rod, etc. which may affect the outcome is avoided to simplify the analysis (e.g. the connecting rod centre of mass is at its centre.). Pin friction and sliding friction are neglected. The energy for pumping of fuel and valve operation is actually transient but is assumed to be a steady phenomenon.

3 Algorithm

At any given instant, there is temperature loss from the gases due to conduction across the walls of the cylinder. This depends on the thickness of the walls. Also, the work done on or by the gases cause a change in temperature. This temperature change leads to corresponding change in pressure of the gas. The pressure of the gases pushes the piston against the elastic forces, which depend on the instantaneous elongation. In fact, the thickness of the cylinder walls also depends on the instantaneous elongation. There is a net force on the piston which is transmitted to the crank via the connecting rod. This force on the crank along with loading torque, the friction torque and its own weight causes a new value of angular acceleration. The angular velocity can be updated based on this value of angular acceleration. The instantaneous value of angular velocity is used to update the crank angle. For the new crank angle, the displacement of piston from IDC, the velocity and the acceleration of the piston can be calculated. The engine is declared to have stopped when angular velocity becomes negative, i.e. when crank rotation reverses. This is checked at every instant. The procedure is repeated a number of times to suit the data requirements, which could be printed or plotted. Every loop furnishes a new set of values for the parameters, which help in predicting the next set.

The procedure has to be modified according to the stroke, the engine is undergoing. During suction, since the gases in the chamber are exposed to the inlet manifold, its pressure and temperature are assumed to remain constant throughout (atmospheric). The compression stroke follows the procedure, unchanged. Just after compression stroke and just before power stroke, the combustion is assumed to take place at constant volume and instantaneously, leading to uniform temperature rise. The power stroke also follows the full sequence. At the beginning of exhaust stroke, the opening of the exhaust valve is assumed to cause instantaneous drop in pressure to atmospheric. This value of pressure is retained throughout the exhaust stroke.

The complications involved in modification of the body of the loop for each stroke can be reduced by splitting it into three modules - one for updating temperature, one for updating pressure & the third for dealing with the forces and motion. If temperature remains constant it can be initialized before the stroke, outside the loop & temperature module can be omitted from the body of the loop. Similar thing can be done for constant pressure.

Initial values of engine variables are set by assuming that the engine is cranked at constant rpm, at no load with a rich fuel air mixture. The engine is loaded after it reaches a respectable speed. The engine could be run over any no of cycles and the performance predicted.

4 Modelling

A. Nomenclature

F_{ij} - Force exerted by link i on link j (N)

F_p - Force Exerted on the Piston (N)

x - Increase in length of the cylinder from its stress-free position (m), i.e. the displacement from IDC

y - Change in Inner Radius (m)

l_c - Length of the connecting rod (m)

Φ - Instantaneous angle, the connecting rod makes with the horizontal (*radians*)

θ - Crank Angle (*radians*)

F_{3h} and F_{3v} - Components of inertia force (N)

- m_c - Mass of the connecting rod (kg)
- m_p - Mass of the piston (kg)
- w - Angular velocity of crank ($radians/s$)
- $\frac{d^2x}{dt^2}$ - Acceleration of the piston (m/s^2)
- $\frac{dx}{dt}$ - Velocity of the piston (m/s)
- r_c - Radius of crank (m)
- α - Angular acceleration of crank (rad/s^2)
- α_c - Angular acceleration of connecting rod (rad/s^2)
- I - Mass Moment of Inertia of Crank (kgm^2)
- P - Pressure at any instant inside the cylinder (Pa)
- T - Inner Temperature (K)
- T_o - Atmospheric temperature (K)
- CV - Calorific Value of Fuel (J/kg)
- a - Air Fuel ratio
- C_v - Specific heat constant of fuel air mixture (J/kgK)
- M - Average molecular mass of the mixture (kg/mol)
- M_a - Molecular mass of air (kg/mol)
- M_f - Molecular mass of the fuel (kg/mol)
- E - Modulus of Elasticity of the Composite Material (Pa)
- K - Thermal conductivity of the material (W/mK)
- r_2 - Inner Radius of Engine Cylinder (m)
- r_1 - Outer Radius of Engine Cylinder (m)
- η_c - Combustion efficiency

B. Equations

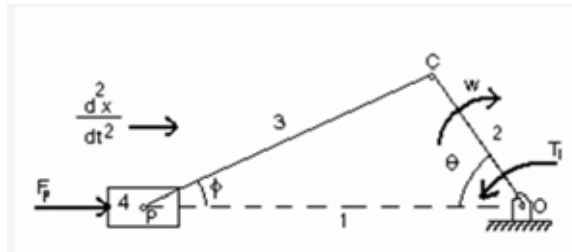


Fig. 1. Basic slider-crank mechanism.

Considering the dynamic equilibrium of the piston gives

$$F_{34h} = F_p - m_p \frac{d^2x}{dt^2} \tag{1}$$

The acceleration of the centre of mass is found by drawing the acceleration diagram. The inertia forces acting on the connecting rod are as follows.

$$F_{3h} = m_c \left[\frac{d^2x}{dt^2} + w^2 r_c \cos \theta + \alpha r_c \sin \theta \right] / 2 \tag{2}$$

$$F_{3v} = m_c (-w^2 r_c \sin \theta + \alpha r_c \cos \theta) / 2 \tag{3}$$

Along the horizontal direction

$$F_{23h} = F_{3h} - F_{43h} \tag{4}$$

Taking moments about P , we get

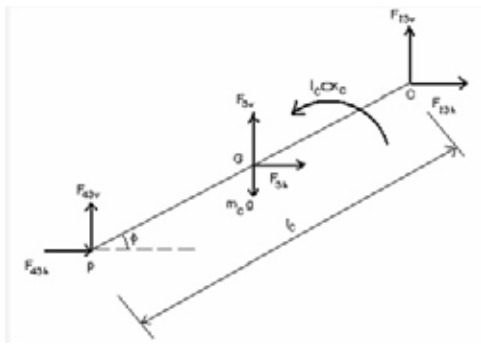


Fig. 2. Free-body diagram of connecting rod.

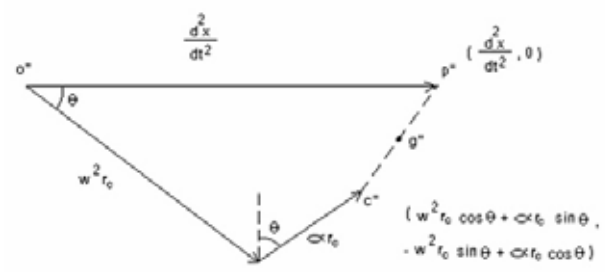


Fig. 3. Acceleration diagram.

$$F_{23v} = F_{23h} \tan \Phi + \frac{I_c \alpha_c}{l_c \cos \Phi} - \frac{F_{3h} \tan \Phi}{2} + \frac{F_{3v} + m_c g}{2} \tag{5}$$

Taking moments about O

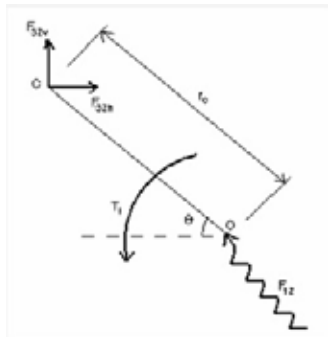


Fig. 4. Free-body diagram of crank.

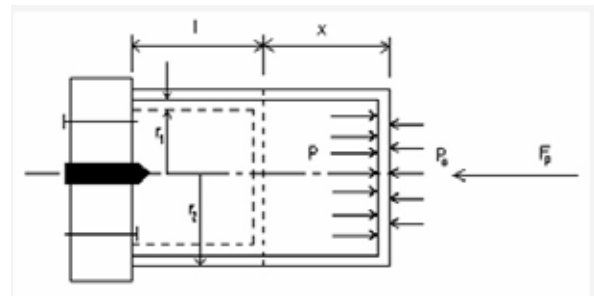


Fig. 5. Free-body diagram of RC engine cylinder.

$$\alpha = \frac{-F_{23h} r_c \sin \theta + F_{23v} r_c \cos \theta + T_l}{I} \tag{6}$$

Balancing the forces acting on the base of the cylinder, we get

$$F_p = (P - P_o) \pi (r_1 + y)^2 - \frac{x \pi (r_2^2 - (r_1 + y)^2) E}{l} - (m + m_{ec}) \frac{d^2 x}{dt^2} \tag{7}$$

The relation between x and y is found by assuming that the material volume remains constant.

$$y = -r_1 + \left| \frac{r_1^2 l + r_2^2 x}{\sqrt{1 + x}} \right| \tag{8}$$

From fig.6, we get,

$$l_c + r + c - l_c \cos \Phi - r_c \cos \theta \tag{9}$$

$$\Phi = \sin^{-1}(r_c \sin \theta / l_c) \tag{10}$$

Differentiating (9) we get

$$\frac{dx}{dt} = \frac{r_c^2}{2l_c} \frac{\sin 2\theta}{\sqrt{1 - (r_c/l_c)^2 \sin^2 \theta}} w + r_c \sin \theta w \tag{11}$$

Differentiating (11) we get

$$\frac{d^2x}{dt^2} = \frac{r_c^2}{2l_c} \frac{1}{\sqrt{1 - (r_c/l_c)^2 \sin^2 \theta}} \left| \frac{1}{2} \frac{w^2 \sin^2 2\theta (r_c/l_c)^2}{(1 - (r_c/l_c)^2 \sin^2 \theta)} + 2w^2 \cos 2\theta + \alpha \sin 2\theta \right| + r_c \alpha \sin \theta + r_c w^2 \cos \theta \tag{12}$$

Differentiating (10) we get

$$w_c = \frac{w \cos \theta}{\sqrt{(l_c/r_c)^2 - \sin^2 \theta}} \tag{13}$$

Differentiating (13), we get

$$\alpha_c = \frac{\alpha \cos \theta}{\sqrt{(l_c/r_c)^2 - \sin^2 \theta}} - w^2 \sin \theta \left| \frac{(l_c/r_c)^2 - 1}{((l_c/r_c)^2 - \sin^2 \theta)^{3/2}} \right| \tag{14}$$

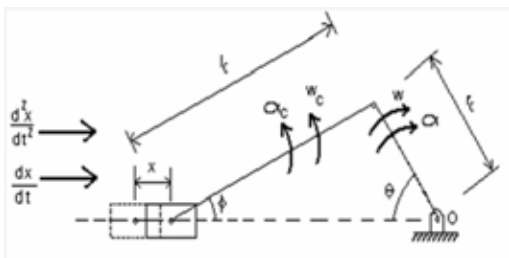


Fig. 6. Kinematic parameters.

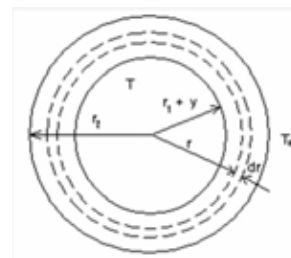


Fig. 7. Radial cross-section of RC engine cylinder.

The gases inside the elastic chamber can be assumed to be a closed system. Thus first law of thermodynamics is applicable.

$$dQ = dU + dW$$

dQ can be found by integrating the Fourier equation for steady state conduction through the cylinder walls, from $r_1 + y$ to r_2

$$\frac{dQ}{dt} = \frac{-2\pi k(1+x)(T - T_o)}{\ln(r_2/(r_1 + y))} \tag{15}$$

The internal energy change can be substituted as $m C_v dT$ where dT is now the temperature change with time, and dW can be substituted with $P dV$. We get,

$$dT = \left| \frac{-2\pi k(1+x)(T - T_o)}{\ln(r_2/(r_1 + y))} - PdV \right| \frac{1}{mC_v} \tag{16}$$

The change in volume in time dt would be

$$dV = 2\pi(r_1 + y)(1+x) \frac{dy}{dx} \frac{dx}{dt} dt + \pi(r_1 + y)^2 \frac{dx}{dt} dt \tag{17}$$

The variation of y with x is known (8). Therefore we can find dy/dx by differentiation.

$$\frac{dy}{dx} = \frac{1}{2 \sqrt{(xr_2^2 + lr_1^2)/(x+1)}} \left| \frac{r_2^2(x+1) - (xr_2^2 + lr_1^2)}{(x+1)^2} \right| \tag{18}$$

The increase in temperature as a result of combustion of fuel in air is found by assuming that most of the fuel available is combusted instantly at constant volume and all the heat generated is absorbed by the mixture. We get

$$\eta_c \frac{m}{a+1} CV = mC_v dT$$

Thus we get

$$dT = \eta_c \frac{CV}{(a+1)C_v} \quad (19)$$

The average molecular mass of the mixture, M is

$$M = \frac{(a+1)(M_a + M_f)}{aM_f + M_a} \quad (20)$$

C. Engine Specifications

Parameters	RC Engine	Nemett IC Engine
Inner radii	0.02 m (at IDC)	0.025 m
Outer radii	0.03 m	0.03 m
Stroke	0.03 m	0.03 m
Clearance	0.00175 m	0.00175 m
Speed	5700 rpm	5700 rpm

The above values were used for the initial size of the engines

D. Simulation

The coding was done in C++. All data were made global. The updation of temperature was done in the function, “temperature()”. The updation of pressure was done in the function, “pressure()”. A third function called ‘dynamics()’ was made to take care of the force transmission and the motion parameters. The function, “dynamics()”, returns 0 when the angular velocity becomes negative and returns 1 otherwise.

The “main()” starts with “first suction” and “first compression”, which happens at a constant rpm. This simulates cranking. After that, there is a loop, the body of which contains one full cycle, i.e. all the 4 strokes. The counter for the loop represents the number of cycles undergone. Each stroke has necessary initializations and iteration. The program ends with the results printed.

5 Result

A. Load test

The load test was conducted by putting arbitrary values for the air fuel ratio and the loading torque, until the speed of the engine steadied to approximately, a constant value (around 5700 rpm). This is how loading & governing was done. There were cases when the speed of the engine increased steadily to very high values and other cases where the engine stopped.

The following characteristic curves of the RC Engine were plotted by conducting a load test in the RC engine simulator.

- (1). TFC vs. BP
- (2). SFC vs. BP
- (3). IP vs. BP
- (4). Mechanical efficiency vs. BP
- (5). Brake Thermal Efficiency vs. BP
- (6). Indicated Thermal Efficiency vs. BP

B. Comparison with IC engine

To do this, we modified the RC engine simulator to work as an IC engine. The modulus of elasticity was made zero. The variations in inner radius were also made zero. The value of thermal conductivity was made higher as the cylinder is made of metal. The dimensions of this IC engine cylinder are also in accordance with the Nemmet 15 cc single cylinder engine.

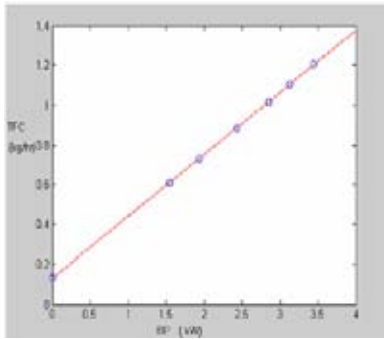


Fig. 8. TFC vs. BP curve.

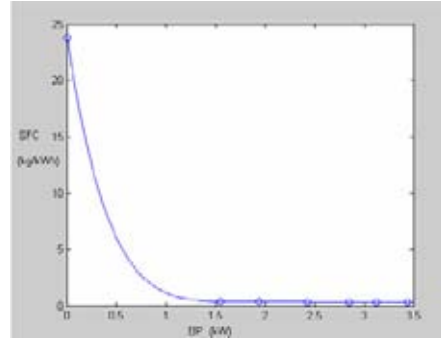


Fig. 9. SFC vs. BP.

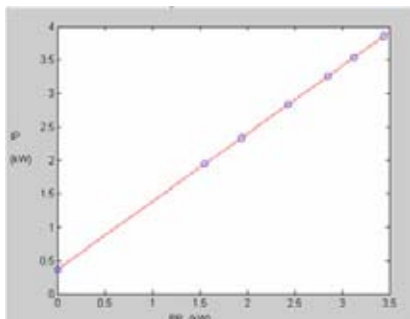


Fig. 10. IP vs. BP.

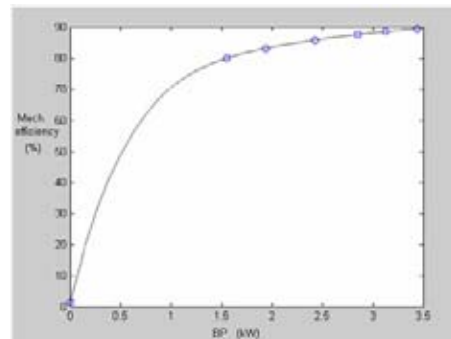


Fig. 11. Mech Efficiency vs. BP.

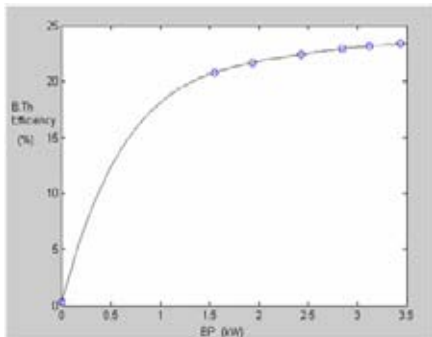


Fig. 12. B.Th Efficiency vs. BP.

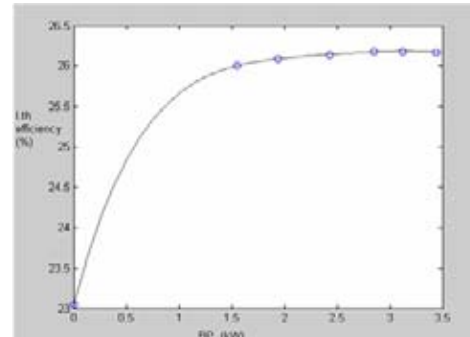


Fig. 13. I.Th Efficiency vs. BP.

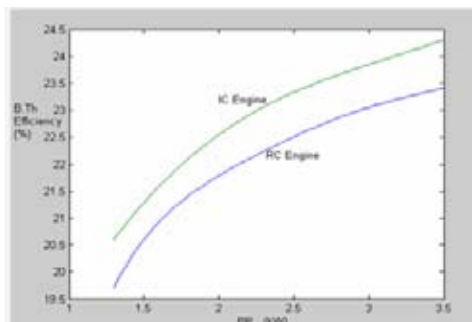


Fig. 14. Comparison of Brake Thermal Efficiency of IC Engine & RC Engine.

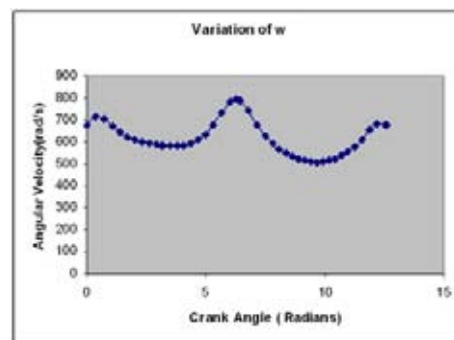


Fig. 15. Variation of w over a cycle.

The characteristic curves were plotted.

(1) The Friction Power was assumed a suitable constant. It didn't increase with loads. As a result we see that that efficiency curves doesn't dip.

(2) High Mechanical Efficiency could also be attributed to this reason.

(3) Brake Thermal Efficiency obtained was in the range of 23% for brake power from 3 kW onwards.

(4) By conducting load test on the IC simulator and RC simulator, we found that efficiency of IC engine is more than RC engine but actually in practical tests conducted, RC engine was found to be more efficient. This could be attributed to the following factors not being taken into account in the IC simulator.

(a) Mechanical Friction that exists in the IC Engine

(b) Knocking has existence in IC but in RC engines, the disturbances are absorbed by the elastic cylinder. Higher Compression ratios were possible for RC engines

(c) piston-cylinder leakage

(5) At the end of exhaust stroke, maximum angular velocity is attained. This is due to fact that elastic energy aids in the return of the stroke. This could also help in efficient removal of the exhaust gases & increase the efficiency.

References

- [1] Balaguruswamy. Object oriented programming using c++. *Tata Mc.Graw Hill*.
- [2] P. Ballaney. *Thermal Engineering in SI units*. Khanna Publishers.
- [3] Incropera, Dewitt. *Fundamentals of Heat & Mass Transfer*. John Wiley & Sons.
- [4] R. Pratap. *Getting Started with Matlab*. Oxford University Press, 2004.
- [5] Ratan. *Theory of Machines*. Tata Mc.Graw Hill.

Appendix

```
#include<iostream.h>
#include<conio.h>
#include<process.h>
#include<math.h>
const float Po = 1.013 * pow( 10 , 5 ) ; // Pa ( Atm pressure )
const float To = 303.15 ; // K ( 30 degree C )
const float l = .00175 ; // m ( clearance length - IDC )
const float rc = .015 ; // m ( radius of crank )
const float lc = .045 ; // m ( length of connecting rod )
const float r1 = .02 ; // m ( initial inner radius )
const float r2 = .03 ; // m ( constant outer radius )
const float pi = 3.1416 ;
const float wo = 1500 * 2 * pi / 60 ; // rad/s ( cranking at 1500 rpm )
float a = 10 ; // 15 kg air for 1 kg fuel
const float Ma = ( .21 * 32 + .79 * 28 ) * pow( 10 , -3 ) ; // m.wt of pure air in kg/mol
const float Mf = 111 * pow( 10 , -3 ) ; // M.wt of fuel in kg/mol ( Petrol )
float M = ( a + 1 ) * Ma * Mf / ( a * Mf + Ma ) ; // M.wt of fuel-air mixture
const float CV = 44000000 ; // J/kg ( CV of Petrol )
const float S = 718 ; // J/kgK ( Sp. heat capacity of air - fuel mixture )
// S for fuel unknown ... S for air at CONST VOL is used
const float k = 0.300124 ; // W/mK ( Thermal conductivity of Silicone )
const float E = 3 * pow( 10 , 6 ) ; // 1 MPa ( Young's modulus )
const float Ru = 8.314 ; // J/molK ( Universal gas const. )
const float dt = 1 * pow( 10 , -5 ) ; // sec. ( differential element of time )
const float mp = 0.5 ; // kg ( mass of piston )
const float mc = 0.5 ; // kg ( mass of connecting rod )
const float I = 3 * rc * rc ; // kg m2 ( Moment of inertia of crank )
```

```

const float Ic = mc * lc * lc / 12 ; // moment of inertia of connecting rod
const float effc = 0.5 ; // combustion efficiency const float effv = 0.75 ; // volumetric efficiency
float eff ; // thermal efficiency float t , x , y , w , alpha , theta , phi , dx_dt , d2x_dt2 , dy_dx ;
// time, elongation of cylinder, variation in thickness of cylinder, angular vel.
//angular acc., crank angle, connecting rod angle, velocity of piston,etc.
float P , T , V , m , dV ; // Pressure,Temp.,Vol.,mass,Vol. element dV
float Fp , F43v , F43h , F23v , F23h , F3v , F3h , alphac ;
// Forces related to connecting rod and angular acceleration of connecting rod
const float Tf = 0.66 ; // Nm ( friction torque )
float Tl = 0 ; // Nm ( loading torque )
int noofc = 0 ; // no. of cycles
int dynamics() // returns 1 on success , 0 on failure
Fp = ( P - Po ) * pi * pow( r1 + y , 2 ) - x * pi * ( r2 * r2 - pow( r1 + y , 2 ) ) * E / 1 - m * d2x_dt2 / 2 ;
// actually inertia force due to elastic chamber also has to be subtracted
d2x_dt2 = ( rc * rc / ( 2 * lc * sqrt( 1 - pow( rc * sin( theta ) / lc , 2 ) ) ) ) * ( w * w * pow( sin( 2 * theta )
* rc / lc , 2 ) / ( 2 * ( 1 - pow( rc * sin( theta )
/ lc , 2 ) ) ) + 2 * w * w * cos( 2 * theta ) + alpha * sin( 2 * theta ) ) + rc * alpha * sin( theta ) + rc * w *
w * cos( theta ) ;
F43h = Fp - mp * d2x_dt2 ;
F3h = mc * ( d2x_dt2 + w * w * rc * cos( theta ) + alpha * rc * sin( theta ) ) / 2 ;
F3v = mc * ( - w * w * rc * sin( theta ) + alpha * rc * cos( theta ) ) / 2 ;
F23h = F3h - F43h ;
alphac = ( alpha * cos( theta ) ) / sqrt( pow( lc / rc , 2 ) - pow( sin( theta ) , 2 ) ) - pow( w , 2 ) * sin( theta
)* ( pow( lc / rc , 2 ) - 1 ) / pow( pow( lc / rc , 2 ) - pow( sin( theta ) , 2 ) , 1.5 ) ;
F23v = F23h * tan( phi ) + F3v / 2 + mc * 9.81 / 2 - F3h * tan( phi ) / 2 + Ic * alphac / ( lc * cos( phi ) ) ;
alpha = - ( F23h * rc * sin( theta ) + F23v * rc * cos( theta ) + ( Tl + Tf ) ) / I ;
w += alpha * dt ;
if ( w < 0 )
{
cout << "\nEngine stopped" ;
getch() ;
return 0 ;
}
theta += w * dt ;
phi = asin( rc * sin( theta ) / lc ) ;
x = lc + rc - lc * cos( phi ) - rc * cos( theta ) ;
y = - r1 + sqrt( ( r1 * r1 * 1 + r2 * r2 * x ) / ( x + 1 ) ) ;
V = pi * pow( r1 + y , 2 ) * ( 1 + x ) ;
return 1 ;
}
void temperature()
{
dy_dx = ( r2 * r2 * ( x + 1 ) - x * r2 * r2 - 1 * r1 * r1 ) / ( 2 * sqrt( ( x * r2 * r2 + 1 * r1 * r1 ) * ( x + 1 ) ) *
( x + 1 ) ) ;
dx_dt = rc * rc * sin( 2 * theta ) * w / ( 2 * lc * sqrt( 1 - pow( rc * sin( theta ) / lc , 2 ) ) ) + rc * sin( theta
) * w ;
dV = 2 * pi * ( r1 + y ) * ( 1 + x ) * dy_dx * dx_dt * dt + pi * pow( r1 + y , 2 ) * dx_dt * dt ;
T += ( ( - 2 * pi * k * ( 1 + x ) * ( T - To ) * dt ) / log( r2 / ( r1 + y ) ) - P * dV ) / ( m * S ) ;
}
void pressure()
{ M = ( a + 1 ) * Ma * Mf / ( a * Mf + Ma ) ; P = m * Ru * T / ( M * V ) ; }

```

```

void main()
{
clrscr() ;
// First suction
w = wo ;
alpha = 0 ;
theta = pi ;
P = Po ;
T = To ;
x = 2 * rc ;
y = - r1 + sqrt( ( r1 * r1 * 1 + r2 * r2 * x ) / ( x + 1 ) ) ;
V = pi * pow( r1 + y , 2 ) * ( 1 + x ) ;
M = ( a + 1 ) * Ma * Mf / ( a * Mf + Ma ) ;
m = effv * P * V * M / ( Ru * T ) ;
// First compression
for( theta = pi ; theta <= 2 * pi ;
theta += w * dt )
{
phi = asin( rc * sin( theta ) / lc ) ;
x = lc + rc - lc * cos( phi ) - rc * cos( theta ) ;
y = - r1 + sqrt( ( r1 * r1 * 1 + r2 * r2 * x ) / ( x + 1 ) ) ;
V = pi * pow( r1 + y , 2 ) * ( 1 + x ) ;
temperature() ;
pressure() ;
}
t = 0 ;
float t100 = 0 ;
for ( noofc = 0 ; noofc < 100 ; noofc ++ )
{
// Combustion stroke
T += effc * CV / ( ( a + 1 ) * S ) ;
for ( theta = 0 ; theta <= pi ; t += dt )
{
pressure() ;
if ( ! dynamics() ) exit( 0 ) ;
temperature() ;
}
cout << endl << w ;
// Exhaust stroke
P = Po ;
for ( theta = pi ; theta <= 2 * pi ; t += dt )
{
if ( ! dynamics() ) exit( 0 ) ;
temperature() ;
}
cout << endl << w ;
// Suction stroke
P = Po ;
T = To ;
for ( theta = 0 ; theta <= pi ; t += dt )
{

```

```

if ( ! dynamics() ) exit( 0 ) ;
}
cout << endl << w ;
// Compression stroke
for ( theta = pi ; theta <= 2 * pi ; t += dt )
{
if ( ! dynamics() ) exit( 0 ) ;
temperature();
pressure();
}
cout << endl << w << endl ;
if ( w <= 523 ) // when rpm < 5000, loading is done
{
Tl = 3.98 ; a = 15 ;
if ( noofc == 98 )
t100 = t ;
}
eff = Tl * 4 * pi * ( a + 1 ) * 100 / ( m * CV ) ;
//cout << endl << "Avg Speed : " << 120 / ( t - t100 ) << " rpm" ;
//cout << endl << "Average ang vel : " << 4 * pi / ( t - t100 ) ;
//cout << endl << "Air fuel ratio : " << a ;
//cout << endl << "Mass of fuel air mixture intake : " << m ;
cout << endl << "Brake Power : " << Tl * 4 * pi / ( t - t100 ) / 1000 << " kW" ;
cout << endl << "Total Fuel Consumption : " << m * 60 * 60 / ( ( a + 1 ) * ( t - t100 ) ) << " kg/hr" ;
cout << endl << "Specific Fuel Consumption : " << ( m * 60 * 60 / ( ( a + 1 ) * ( t - t100 ) ) ) / ( Tl *
4 * pi / ( t - t100 ) / 1000 ) << " kg/kWh" ;
cout << endl << "Indicated Power : " << ( Tl + Tf ) * 4 * pi / ( t - t100 ) / 1000 << " kW" ;
cout << endl << "Mechanical efficiency : " << Tl / ( Tl + Tf ) * 100 ;
cout << endl << "Brake Thermal Efficiency : " << eff ;
cout << endl << "Indicated Thermal Efficiency : " << ( Tl + Tf ) * 4 * pi * ( a + 1 ) * 100 / ( m * CV
); getch();
}

```