bool is\_in\_reference\_domain(const double xi)

{

if (xi < -1.0 || xi > 1.0)

{

return false;

}

else

{

return true;

}

}

void CPnload::CalcQuadNaturalCoord(double & eta,double & xi,const double xp,const double yp,

const double x1,const double x2,const double x3,const double x4, const double y1,const double y2,const double y3,const double y4)

{

//f1 = shape\_function1\*x1 + ... + shape\_function4\*x4 - xp

// = a\_0 + a\_1 \* xi + a\_2 \* eta + a\_3 \* xi\*eta;

const auto a\_0 = 0.25 \* (x1 + x2 + x3 + x4) - xp;

const auto a\_1 = 0.25 \* (-x1 + x2 + x3 - x4);

const auto a\_2 = 0.25 \* (-x1 - x2 + x3 + x4);

const auto a\_3 = 0.25 \* (x1 - x2 + x3 - x4);

//f2 = shape\_function1\*y1 + ... + shape\_function4\*y4 - yp

// = b\_0 + b\_1 \* xi + b\_2 \* eta + b\_3 \* xi\*eta;

const auto b\_0 = 0.25 \* (y1 + y2 + y3 + y4) - yp;

const auto b\_1 = 0.25 \* (-y1 + y2 + y3 - y4);

const auto b\_2 = 0.25 \* (-y1 - y2 + y3 + y4);

const auto b\_3 = 0.25 \* (y1 - y2 + y3 - y4);

//a xi^2 + b xi + c = 0

const auto a = a\_1 \* b\_3 - a\_3 \* b\_1;

const auto b = a\_1 \* b\_2 - a\_2 \* b\_1 + a\_0 \* b\_3 - a\_3 \* b\_0;

const auto c = a\_0 \* b\_2 - a\_2 \* b\_0;

const auto discriminant = b \* b - 4 \* a \* c;

if (discriminant < 0)

{

std::cout << "Negative discriminant Bug! \n";

return;

}

const auto temp1 = std::sqrt(discriminant);

const auto temp2 = 1.0 / (2 \* a);

const auto xi\_1 = (-b + temp1) \* temp2;

const auto xi\_2 = (-b - temp1) \* temp2;

if (is\_in\_reference\_domain(xi\_1))

{

xi = xi\_1;

if (a\_2 + a\_3 \* xi\_1 == 0)

{

eta = -(b\_0 + b\_1 \* xi\_1) / (b\_2 + b\_3 \* xi\_1);

}

else

{

eta = -(a\_0 + a\_1 \* xi\_1) / (a\_2 + a\_3 \* xi\_1);

}

}

else if (is\_in\_reference\_domain(xi\_2))

{

xi = xi\_2;

if (a\_2 + a\_3 \* xi\_2 == 0)

{

eta = -(b\_0 + b\_1 \* xi\_2) / (b\_2 + b\_3 \* xi\_2);

}

else

{

eta = -(a\_0 + a\_1 \* xi\_2) / (a\_2 + a\_3 \* xi\_2);

}

}

else

{

std::cout << "Out of range Bug!\n";

}

}