# Euler / Improved Euler

```
void simulation::UpdateEuler(double h){
	ClearAccelerations();

	for (forces.MoveFirst(); !forces.AtEnd(); forces.MoveNext())
		forces.GetCurrent()->CalculateForce();

	DivideByMass();

	HandleFixed();
	for (particles.MoveFirst(); !particles.AtEnd(); particles.MoveNext()){
		particle *p = particles.GetCurrent();

		p->position += p->velocity * h;
		p->velocity += p->acceleration * h;
	}
	HandleGround();
}

void simulation::UpdateImprovedEuler(double h){
	particle *part;

	// Calculate the acceleration now
	ClearAccelerations();
	for (forces.MoveFirst(); !forces.AtEnd(); forces.MoveNext())
		forces.GetCurrent()->CalculateForce();
	DivideByMass();

	HandleFixed();

	// Store the positions, velocities and accelerations
	int i;
	for (particles.MoveFirst(), i = 0; !particles.AtEnd(); particles.MoveNext(), i++){
		part = particles.GetCurrent();
		p[0][i] = part->position;
		v[0][i] = part->velocity;
		a[0][i] = part->acceleration;
	}

	// Calculate the position after h seconds
	for (particles.MoveFirst(); !particles.AtEnd(); particles.MoveNext()){
		part = particles.GetCurrent();
		part->position += part->velocity * h;
		part->velocity += part->acceleration * h;
	}

	// Calculate the acceleration there
	ClearAccelerations();
	for (forces.MoveFirst(); !forces.AtEnd(); forces.MoveNext())
		forces.GetCurrent()->CalculateForce();
	DivideByMass();

	HandleFixed();
```

# Euler / Improved Euler

```
        // Use the averages of the velocity and acceleration to
        // update the original p's and v's
        for (particles.MoveFirst(), i = 0; !particles.AtEnd(); particles.MoveNext(), i++){
                part = particles.GetCurrent();
                part->position = p[0][i] + (v[0][i] + part->velocity) * h / 2;
                part->velocity = v[0][i] + (a[0][i] + part->acceleration) * h / 2;
        }
        HandleGround();
}
```