

C++ code 5.15.25: C++ Code for function curve_points()

```
2 Eigen::Matrix2Xd ClosedQuadraticSplineCurve::curve_points(  
3     const Eigen::VectorXd &v) const {  
4     unsigned int N = v.size();  
5     assert(N > 0);  
6     // Matrix containing points to be computed  
7     Eigen::Matrix2Xd s(2, N);  
8     // Lengths of knot intervals  
9     const Eigen::VectorXd h = t_.tail(n_) - t_.head(n_);  
10    // TODO: (5-15.d) Compute the points on the curve for the sorted  
    // parameter  
    // values v.  
    // START  
11  
12  
13  
14    // Run through all the provided parameter values  
15    unsigned int knot_idx = 0;  
16    for (unsigned int k = 0; k < N; ++k) {  
17        assert(((v[k] >= 0.0) && (v[k] < 1.0)) && "Out of range!");  
18        if (k > 0) {  
19            assert(v[k] >= v[k - 1] && "Parameter values not sorted!");  
20        }  
21        // Find right knot interval: knot_idx stores index of knot to the  
        // right of  
        // current parameter value  
22        while ((knot_idx < n_) && (v[k] >= t_[knot_idx])) {  
23            knot_idx++;  
24        }  
25        assert(knot_idx > 0);  
26        const double tau = (v[k] - t_[knot_idx - 1]) / h[knot_idx - 1];  
27        s.col(k) = ((1.0 - tau) * p_.col(knot_idx - 1)) +  
28            (h[knot_idx - 1] * tau * (1 - tau)) * x_.col(knot_idx - 1) +  
29            (tau * p_.col(knot_idx));  
30    }  
31    // END  
32    return s;  
33 }  
34 }
```

Get it on 🐘 [GitLab](#) (periodicquadraticsplines.hpp).