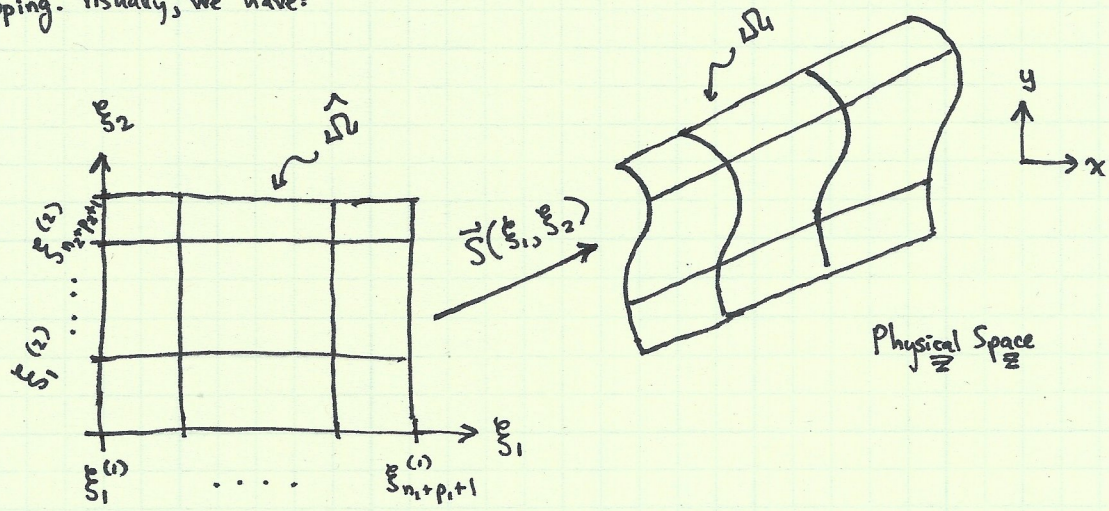
Multi-Patch Geometries:

In almost all practical circumstances, it is practically impossible to represent the geometry of interest with a single NURBS mapping. Thus, it becomes necessary to describe the geometry using multiple NURBS mappings. We refer to such geometries as multi-patch geometries.

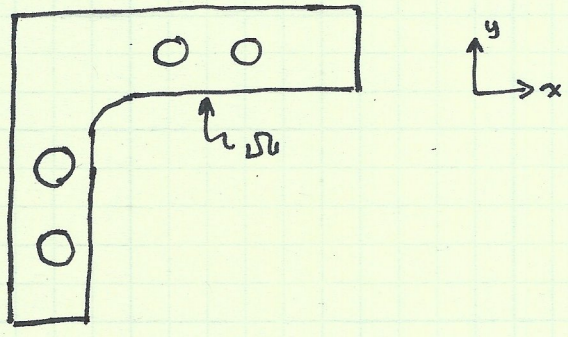
To describe what we mean by "patch", we return to the setting of a single NURBS mapping. Visually, we have:



Parametric Space

The single NURBS mapping $\vec{S}(\xi_1, \xi_2)$ takes the patch from parametric space to physical space. We interchangeably refer to $\hat{\Omega}$ and its image Ω as the NURBS patch. Note that for a single-patch geometry, the patch Ω is the domain.

Now consider the following domain:



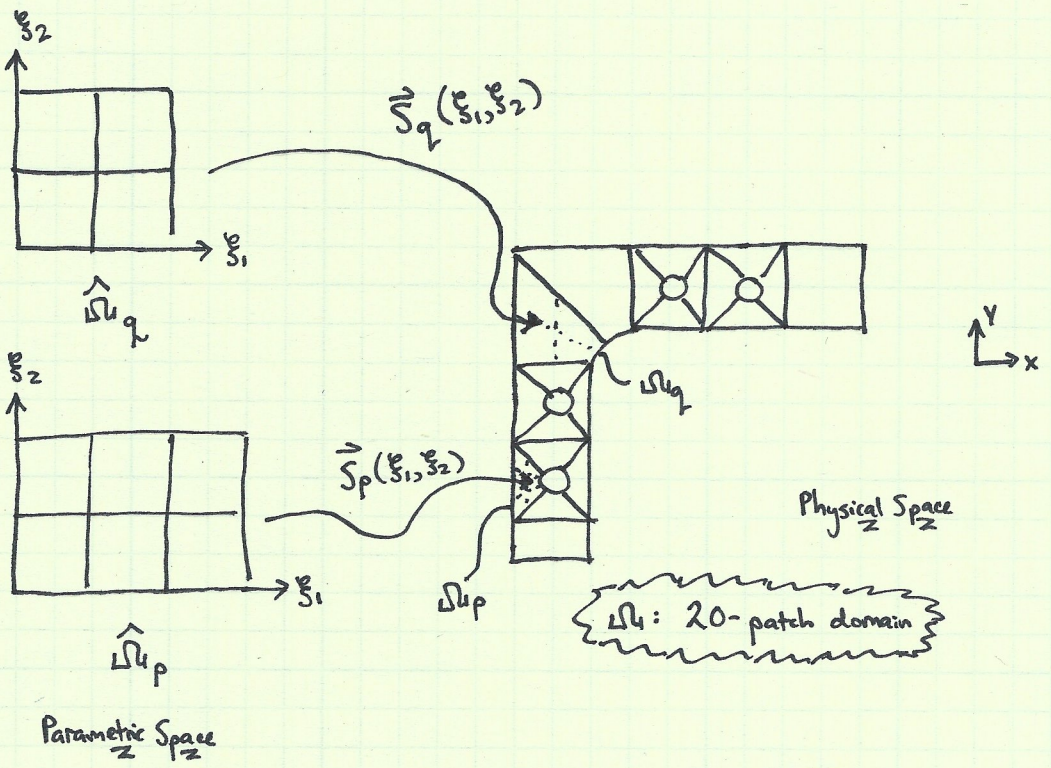
For this configuration, it is impossible to use a single NURBS map to parameterize the geometry. Instead, we must use multiple NURBS patches and mappings:

$$\vec{S}_p(\xi_1, \xi_2) : \hat{\Omega}_p \rightarrow \Omega_p \qquad p = 1, \ldots, n_{patch}$$

$\hat{\Omega}_p$: Patch p in parametric space

Ω_p: Patch p in physical space

Visually:



To ensure a watertight geometry, we must ensure the patches $\{\Omega_p\}_{p=1}^{n_{patch}}$ meet parametrically and geometrically on the internal faces where they meet.

To be more precise, let Ω_i and Ω_j be two disjoint patches such that $\partial\Omega_i \cap \partial\Omega_j \neq \emptyset$. Then, we require that $\vec{S}_i$ and $\vec{S}_j$ parametrize the shared face $\partial\Omega_i \cap \partial\Omega_j$ exactly up to changes in orientation. This requires that the control nets associated with $\partial\Omega_i$ and $\partial\Omega_j$ are coincident. ↳ and identically!

All of the aforementioned Bézier extraction machinery may be easily extended to the multi-patch setting. All we need is a mechanism to associate patch element numbers and basis function numbers to global element numbers and basis function numbers. We use the following two arrays in this regard:

$$\text{PTG element}\,(p, e_p) = e$$

Global Element Number

Patch Number — Local Element Number

$$\text{PTG basis}\,(p, i_p) = i$$

Global Basis Function Number

Patch Number — Local Basis Function Number

To construct these arrays, we order elements and basis functions patch-by-patch. That is, the elements associated with patch 1 have global element numbers $1, \ldots, n_{el_1}$, where

n_{el_p} is the number of elements associated with patch 1, the elements associated with patch 2 have global element numbers $n_{el_1}+1, \ldots, n_{el_1}+n_{el_2}$, and so forth. As control nets associated with shared faces must be coincident, we also mate basis functions at shared interfaces. This requires a master-slave relationship between such basis functions. If Ω_i and Ω_j are two disjoint patches with $\partial\Omega_i \cap \partial\Omega_j \neq \phi$, we enforce that Ω_i is slave to Ω_j if $i > j$ and vice versa.

With the above conventions defined, we are ready to present a function which, given the number of elements n_{el_p} and number of basis functions n_p associated with each patch, returns the PTG element and PTG basis arrays:

```
function Patch Connectivity ( {n_elp}_{p=1}^{n_patch} , {n_p}_{p=1}^{n_patch} )
begin
    Set itr = 0

    for p = 1, ..., n_patch
        for e_p = 1, ..., n_elp
            set PTG element (p, e_p) = e_p + itr        <- Construct PTG element
        endloop
        Set itr = itr + n_elp
    endloop

    Set itr = 0

    for p = 1, ..., n_patch
        for i_p = 1, ..., n_p
            Set flag = 0
            Set q = 1
            while (q < p)
                for i_q = 1, ..., n_q
                    if P_{i_p,p} = P_{i_q,q}        <- Control Point i_p of Patch p; Control Point i_q of Patch q

                        Set PTG basis(p, i_p) =
                                PTG basis (q, i_q)      <- Construct PTG basis
                        Set flag = 1
                        break out of while loop

                    endif
                endloop
            endloop
            if flag = 0
                Set PTG basis (p, i_p) = itr + 1
                Set itr = itr + 1
            endif
        endloop
    endloop

    return PTG element and PTG basis
end
```

Using the PTG element and PTG basis arrays, we can now easily compute the element extraction operators and IEN array for the global domain from the patch-wise element extraction operators and IEN arrays. We restrict our attention to the 2D setting.

function Extract And Localize $\left(\{ n_{1p} \}_{p=1}^{n_{patch}}, \{ n_{2p} \}_{p=1}^{n_{patch}}, \{ p_{1p} \}_{p=1}^{n_{patch}}, \{ p_{2p} \}_{p=1}^{n_{patch}}, \{ \Xi_{1p} \}_{p=1}^{n_{patch}}, \{ \Xi_{2p} \}_{p=1}^{n_{patch}} \right)$

begin

for $p = 1, \dots, n_{patch}$

Call Extraction 2D $(n_{1p}, n_{2p}, p_{1p}, p_{2p}, \Xi_{1p}, \Xi_{2p})$ to obtain:

- n_{el_p} : The number of elements of patch p
- $\{ \underline{\underline{C}}_p^{e_p} \}_{e_p=1}^{n_{el_p}}$: The element extraction operators for patch p

Call IEN 2D $(n_{1p}, n_{2p}, p_{1p}, p_{2p}, \Xi_{1p}, \Xi_{2p})$ to obtain the IEN array for patch p, which is denoted here as IEN_p

Set n_p as the maximum entry of IEN_p

endloop

Call Patch Connectivity $\left(\{ n_{el_p} \}_{p=1}^{n_{patch}}, \{ n_p \}_{p=1}^{n_{patch}} \right)$ to obtain

- PTG element
- PTG basis

for $p = 1, \dots, n_{patch}$

for $e_p = 1, \dots, n_{el_p}$

Set e = PTG element (p, e_p)
Set $\underline{\underline{C}}^e = \underline{\underline{C}}_p^{e_p}$

Extraction Operators

for $a = 1, \dots, n_{loc_p} = (p_{1p}+1)(p_{2p}+1)$

Set $i_p = IEN_p(a, e_p)$
Set i = PTG basis (p, i_p)

Set $IEN(a, e) = i$

IEN Array

endloop

endloop

endloop

return $n_{el} = \sum_{p=1}^{n_{patch}} n_{el_p}$, $\{ \underline{\underline{C}}^e \}_{e=1}^{n_{el}}$, and IEN

end

It should be noted that even in cases where a single-patch geometry may be employed, doing so might introduce extreme mesh distortion which may adversely affect finite element analysis. The picture below demonstrates such a situation. Alternatively, a multi-patch geometry may introduce far less distortion, leading to improved finite element results.