A Computation Optimization Method in Computer Mediated Teleoperation

J. K. Mukherjee,
EISD, Bhabha Atomic Research Centre,
Mumbai, 400085, India

ABSTRACT

In modern master slave system, which have computer mediator between the two sides, the mediator computer faces high and non deterministic computational load arising from complex problem of computing instantaneous physical distances of moving robot from the modeled objects in the workspace. The problem is acute for mediators that work in pre-contact state where tool-tip is in close vicinity to object body. A novel vicinity model based method has been developed to minimize real time computation load. Fixed quantum, minimized floating point computation attains deterministic real time performance while mediating in force feedback modality.

General Terms

Tele-operation, Kinematics model, computer mediation, CAD models.

Keywords

Computation load optimization, spatial state, vicinity, tele-robot, time-critical computation, mediator computer.

1. INTRODUCTION

Tele-controlled Robotic arms working as slave agents under human operator control are versatile remote working tool in hot cells that constrain the viewing highly and make other means of perceiving conditions in workspace more desired. Bilateral manipulators offer transparency by force feedback on contact only [1]. The phenomena supported are limited to those occurring in contact phase and some effects from high loads accruing from high acceleration, high velocity and robot spatial configuration dependant gravity-load variations. Under no-contact state velocity based damping is also offered [2]. But it does not depend on workspace neighbourhood. Also stability related difficulties exist [3] [4].

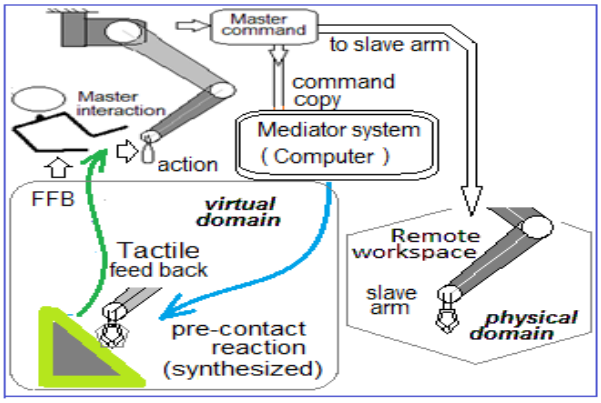


Figure 1. Computer mediated system

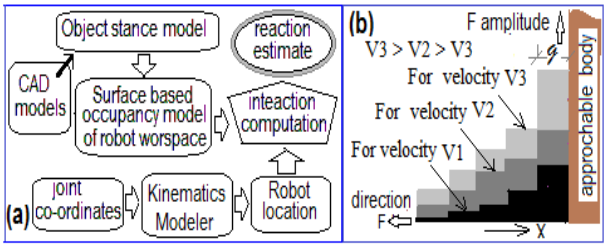


Fig.2a Pre-contact reaction synthesis modality using CAD modelled workspace, 2 b. desired feedback in object vicinity.

Limited perception of workspace causes erroneous operation leading to high probability of hits. Graphics based method has been attempted to provide predictive display [5]. Robot approaching a workspace must be discouraged to continue the approach. Method for creating system reaction by on line estimation of closeness to obstacle must work in real time.

Computer mediated system (fig.1) can generate synthesised reaction to operator of tele-controller master. Computation based synthesis of reaction between the object models and robot model is done by taking an interaction computation route that uses surface based occupancy model and robot kinematics model (fig 2). The approach rebuilds the state of the remote robotic arm in modelled workspace by monitoring the tele-robot's joint parameters. By maintaining synchronism of the modelled robot state with tele-robot, the mediator computer emulates relative spatial state of a robot and workspace in virtual domain at same time as the real domain tele-robot. The synthesized reaction must be dependent on degree of closeness to an object and robot dynamics. Use of CAD based part models in STL format [6] for forming workspace model permits automation and so is a natural choice. Though for very simple objects with few surfaces, the direct computation method works, it fails for a workspaces with complex objects or large number of objects. The present work develops an alternative method with bounded time computation suitable for real time operation.

2. COMPUTATION COST

Consider a workspace with object A, B and C and a point robot $\mathcal{R}$ (fig. 3). Surface based occupancy model is made from CAD models or from range sensed data [7]. These comprise triangular surface segments typically in STL version [8][9]. $\mathcal{R}$ must be made sensitive to vicinity of objects in some way. Direct use of physics principles need that physical distance from surface be computed when $\mathcal{R}$ is in vicinity of an object. Vicinity is defined by a zone engulfing the surface on its outer face up to a distance 'δ' from surface (green zones). Though $\mathcal{R}$ has to react within 'δ' distance only from object for vicinity based sensitivity development, the physical distance 'd' from object's surface needs to be computed at any new location of $\mathcal{R}$ to sense whether its present location

is in vicinity of object. Objects comprise many surfaces and so the vicinity test needs to be done with all the surfaces of all the objects in workspace. This may run in thousands. Estimation of *'d'* is computationally costly and time consuming. Since the number of components in a real workspace is variable and the object shape complexity too can change on large range the computation demand is non deterministic

2.1 Improving computation efficiency

Fast procedures have received attention [10], [11]. Computation reduction by excluding objects (and their component surfaces) that are far from $\mathcal{R}$ through simple test for presence of $\mathcal{R}$ within the smallest box (aligned with Cartesian workspace coordinate) formed to fully include vicinity extent of the object is effective (fig 3b). Oversized sphere of radius 'r' too may be used by computing single point distance of $\mathcal{R}$ to the center of the bounding sphere (fig. 3c) and comparing d with 'r'. Spherical object representation have received attention earlier too [12]. Since the non inclusion of a $\mathcal{R}$ position (x,y,z) is tested by simple logical checks at low computation cost (table1 cases1 and 3), in a significant volume (shown as blue in fig. 3d), the computation is light and test is complete within real time limits (5ms in 200 Hz up-date control loop by 1.5 GHz Intel core2 processor for typical workspace with 30- 50 objects). For complex single object, the excessive computation problem persists. For object **B,** the inclusion zone too, has significant volume as depicted by yellow zones in fig 3e and 3f where distance to surfaces of **B** has to be determined by computational geometry methods (fig. 4) and shortest of them needs to be found by a comparison process to find nearest object and **δ**. The nearest surface finding process requires each triangular surface to be considered for computing its unit normal, computing a vector parallel to the normal from a given $\mathcal{R}$, computing its intersection with surface, finding length of this vector and comparing these lengths for all surface triangles to find the shortest of all. Entire process has to be repeated on location change of $\mathcal{R}$.

Use of methods shown in fig 3 can be seen as filters (fig5) that terminate the test loop early and save time. 'T' represents time within which sensing should be complete and procedure should start for the next position of $\mathcal{R}$. Time t_R is available to react to vicinity. For reducing computations, reusable data like 'bounding box extents, centre and radius of bounding spheres etc are computed once at start up and associated with part models as additional persistent data. Filters succeed when $\mathcal{R}$ is away from objects and possibility of intersection is less. Most of the cases encounter fail 1or fail2 As $\mathcal{R}$ progressively reaches closer to object, initial tests are passed till greater depth into the test (fig 3 e, f) and late fail occurs. Intersection failure case (fig.4c) is an example. Consequently, near the object, vicinity is detected at last stage by incurring high computational burden leaving lesser t_R. The gain from the filtering conditions is that candidate population for test reduces in a sequential processor based system and for parallel processor systems a freed processor can be assigned to new $\mathcal{R}$ on full body. The filters are formulated such that their processed data is usable by subsequent filter so that time spent in test is minimized for the whole chain. Search based hit detection by search ordered optimally to achieve fast hit detection have been attempted earlier [13].

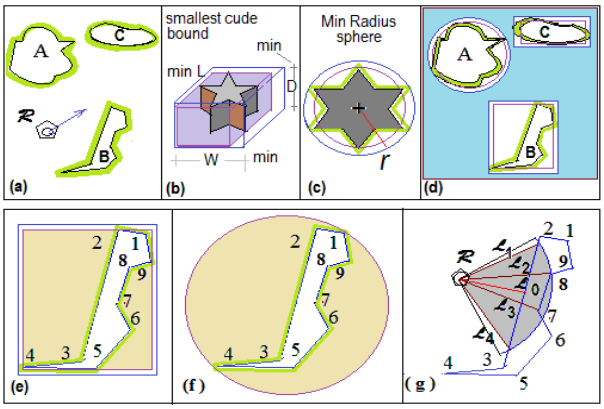


Figure 3 a. Workspace in 2D; b and c. search zone reduction by smallest bounding cube and sphere; d. search zone reduction represented by blue in overall workspace e volume in which robot has to compute surface distance in spite of using bounding box test f. relatively worse condition for sphere test e. distance to surface computation

2.2 Effectiveness of optimization

Fool proof filters do not exist. Non-effectiveness of the filters is indicated by the extent of yellow zones in fig 3e and 3f . Some tests fail in some conditions. In close vicinity the trueness of vicinity is tested by being within 'δ'. Since smallest 'δ' is most dominant in effecting $\mathcal{R}$, nearest STL triangle has to be found. For example attempts to reduce candidate surfaces belonging to a STL modelled object from the complex computational test by an approach based on distances of **p** from nodes $\mathbf{s_1}$, $\mathbf{s_2}$, $\mathbf{s_3}$ forming the surface **S,** fails in some cases. For simplified depiction, 2D plan views is shown in fig 3g. Segments 1-2, 2-3 , 3-4 etc. represent planar surface patches of STL model of object **B.** $\mathcal{L}_1$ to $\mathcal{L}_4$ represent distances from robot '$\mathcal{R}$' to the surfaces. The nearest surface patch 2-3 is formed by nodes 2,3 which are relatively at large distance from $\mathcal{R}$. The surface formed by nearer node set 7-8 has distance $\mathcal{L}$ 0 where $\mathcal{L}$ 0 > $\mathcal{L}$2 and $\mathcal{L}$3. A node distance based criteria must not be used to decide candidature of a surface **S** for computing nearness. For example out of segment 2,3 and 7,8 (fig. 3g) candidature of 7,8 for distance measure may be eliminated by testing surface direction , which should be towards $\mathcal{R}$ for the segment to be considered for assessing distance from $\mathcal{R}$. This test needs considerable computation (table-1 case 7). Problem of finding minimum $\mathcal{L}$ escalates with increase of part count in workspace and also with fine surface definition of quadric surfaces and contoured bodies (fig 5). Work in recent times towards computer mediation have attempted to develop methods for either guiding a robot to preferred spaces by constraining their movement using software implemented techniques or by creating an

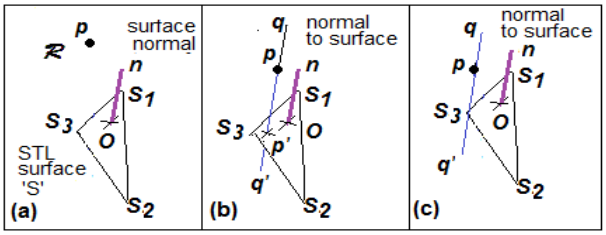


Fig. 4a. STL surface S, b. normal distance *pp'* c. no intersection

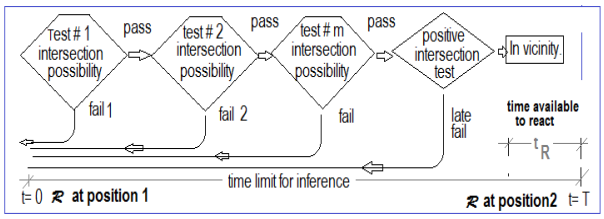
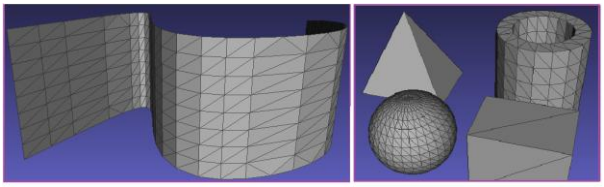


Fig. 5. Optimization by early test termination



. Fig. 6. CAD models using bounding surfaces and planar surface triangles on them for defining object.

opposition to movement towards obstacle by virtual barrier. In order to solve the unbounded computation problem arising from objects of complex shape (fig. 6), planar barrier surfaces of limited number are artificially placed as boundaries around object clusters that are not approachable. The role of the barrier is to reduce computation [14][15]. The drawback is that user has to place these barriers. Improper barrier placement may result in severely compromised workspace situations where free space for robot movement is too constrained [16].

3. COMPUTATION REDUCTION

Preceding attempts of optimisation are robot centric in nature and show that computation reduction is feasible by attaching additional attributes to the object like bounding volumes of simple shapes. While the effectiveness is high when robot is in. 'blue zones' in fig 3 , in significant volumes particularly near the objects e.g. yellow zones in fig 3e and 3f , no benefit may accrue from the optimisation efforts of this type. There is a need for different optimisation method to achieve real time performance reliably irrespective of shape of object. Recent reported work [17] and [18] encourage to take a different look at the problem by using object centric route . Optimised 'vicinity formation with good fidelity and gradation is feasible. The vicinity , depicted by green zone on surfaces, can be treated as object property.

Table 1: Estimated computations for various tests

	Test	Operation Performed		No of op.
1	Within Object sphere	a) Non optimized	(3 S, 3 E, 2A , 1 SR, 1 C)	(9~10)* N_{ob}
2		b) Optimized	(3 S, 3 E, 2 A, 1 C)	
3	Within Surface sphere	a) Non optimized	(3S, 3E, 2A 1 SR, 1 C)	(9~10)* N_{tri} in object
4		b) Optimized	(3S, 3E, 2A 1 C)	
5	object cube	3X(2C +1 L)		M*N_{ob}
6	surface cube	3X(2C +1 L)		M* N_{tri} in object
7	Surfaces Facing Observer	(3 S, 3 M, 2 A, 2 SR, 1 D, 1 T, 1 C)		M*N_{norm}
8	Ray Intersection (N_{ray})	(8 M, 6 A, 1C) ~(12 M, 9 A, 1 D, 1C)		(15~23)* N_{ray}
9	Vector Intersect	(33M, 18S, 19A, 3SR, 2C)		(74~75)* N_{ve}

In table above: S=Subtraction, A=Addition, E=Exponent, SR=Square Root, M=Multiplication, T=Trigonometric, C=Comparator, D=Division N_{ob} is number of objects within CG, N_{tri} is number of triangles within the surface CG, N_{norm} is number of triangles facing ℛ , N_{ray} is number of triangles having extended plain intersection with qq', N_{vec} is triangles intersected by qq'. SR is avoided in cases 1 and 2 by comparing squired operators themselves.

3.1 Fluidics inspired Model

An approach based on fluid flow in a closed piston-cylinder assembly in which cylinder has a set of side branch tubes for the fluid to flow back to its head-end when piston moves from head end towards tail-end (fig.7a), has been suggested recently [19]. As fewer branches remain for flow back with progression of piston, opposition force increases as piston moves towards tail end.

$$F = V * C * (1/ (1/\rho_1 + 1/\rho_2 + 1/\rho_3 \ \ldots \ 1/\rho_n)) \quad \ldots\ldots\ldots(1)$$

Where V is speed of piston move and C is $(D/d)^2$. 'D' and 'd' being diameter of cylinder and bypass respectively (fig.1A) is a constant. Index 'n' varies from 1 to N. $(1/\rho_n)$ is admittance of n^{th} bypass depending on 'l' 'd' and viscosity 'μ' of fluid.

$$\rho = ((128*\mu* l)/\pi* d^4)* (\nu*\pi*D^2)/4 \ \ldots\ldots\ldots\ldots(2)$$

The force model so developed (fig 7b) can be represented as graded bar (GB) with γ values which are the F values in respective segments seen by a point robot passing with unit velocity (fig. 7c). An approaching robot has to push the virtual piston to reach an object in its workspace and so faces opposition force that varies with distance remaining to tail end and its velocity V. It is computed by using relation-

$$F_{(n)} = V * \gamma_{(n)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

User defined formulation of opposition-force 'F' is easy by mere change of dimensions 'n' 'l' 'd' and 'g'

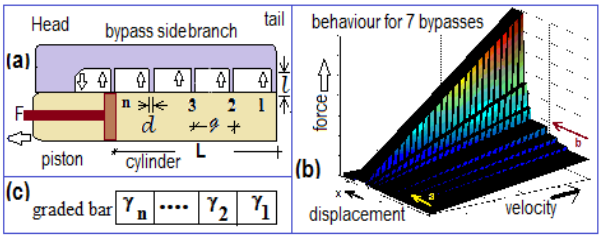


Fig.7 a; multiple pass cylinder b; fluid reaction force on piston. c; coded bar with equivalent F values at unit velocity

3.2 Model adaptation in real work scenario

A robot approaching an object uses a data array representing the grade bar by attaching it as normal vectors to the planar surface patch of the part model in the workspace (fig 8a) along which γ varies. Tail end of GB touches the surface. Attachment of the graded bar as normal to an object surface at its tail end (fig. 8a), poses difficult problem in real time implementation. Consider a point robot instance P near the object shown in fig. 8b. For depiction, 2D plan view are shown. For P, nearest point on object is S at distance 'ℓ'.

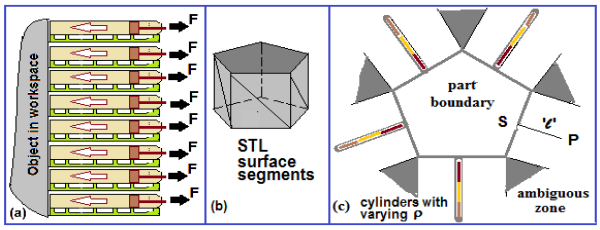


Figure 8a. pentagon cylinder in STL format (b) graded bar as vector attached to flat faces (c) ambiguous zones

.

For activation of piston action, 'ℓ' must be < L i.e. cylinder-length. Computation of 'ℓ' involves finding the closest surface at instantaneous location 'P'. Computation problem is same as described earlie . Reduction of computation at run time can be attempted by preparing all surface data along with the 'graded bar' data at workspace modelling time and only searching the nearest surface patch possessing the data at robot motion execution time. But searching the appropriate surface patch also proves prohibitively time demanding for workspace with complex shaped objects as in fig. 6 and even large population of simple components. Consequently real time proximity sensing can not be ensured. In addition this form of data has ambiguity in the shaded regions (fig 8c).

4. EFFICIENT MODEL APPLICATION

A rearrangement of parallel graded bars on surfaces (fig 8a) on object leads to the surrounding space appearing as in fig. 9.

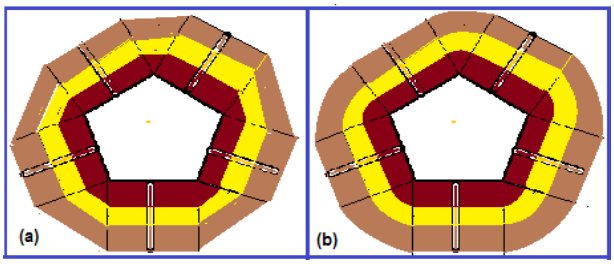


Fig9 : graded neighbour a. linear interpolation, b circular interpolation

Depending on the method applied for interpolating orientation of the cylinders on edge or vertices of object (shaded zones in fig 8c) such as linear , circular and spherical interpolations , resulting arrangement appears as graded neighbour. (fig. 9a, 9b). The object surrounding is now layers of ρ values. This encourages a different look at the problem in hand.

4.1 Voxel based modeling

Voxel oriented approaches have delivered very encouraging results in 3D graphics [20][21]. In such approach [19] [22][23], workspace is represented as 3 dimensional array of equal sized cube shaped volume elements with integer coordinate locations (X,Y,Z) that are referred as 'Voxel' and

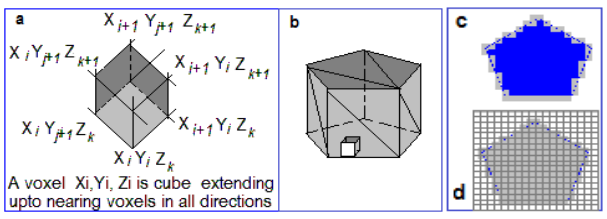


Figure 10a. voxel definition, 10b surface voxel of cylinder object 10c top layer surface voxel and 10d intermediate layer in cylinder with body voxel.

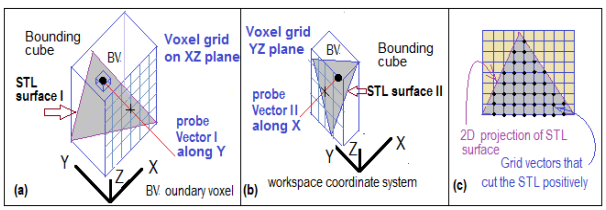
denoted as $V_{(XYZ)}$ in further descriptions. A voxel $V_{(XYZ)}$ represents cube with diagonal nodes X_i,Y_j,Z_k and X_{i+1}, Y_{i+1} Z_{i+1} as shown in figure 10a.The indices range from 0 to a max integer value suitable for representing maximum extent of workspace with desired granularity. The voxels was assigned 'potential' property for path finding problem in [18]. Here we use them to assign other properties. For representation of spatial occupancy of space by parts in the workspace, we adhere to the following interpretation. A grid position in workspace where no object exists is represented by free voxel. A voxel through which a triangular surface passes is boundary voxel (fig. 10b), a voxel inside object is occupied voxel. Typical voxel data is a record (fig. 11) with last field as signed integer. Its use will be evident later. For CAD models in STL version, data representing the 3D shape of object is associated to workspace geometry as per its pose and position in work space using node set rotations and translation. This also eliminates hole forming problem that can occur if voxelised model is rotated [24]. Entire workspace voxel array is initialized as 'empty' type. In phase I, triangular surface segments in CAD- STL model or mesh coded surface triplets from range based volume occupancy models [16] need to be computationally intersected by grid lines of Cartesian coordinate system over the entire workspace range .

Grid Pt. X,Y,Z	Object index	Type (empty / boundary/ inner)	Parameter value (order)

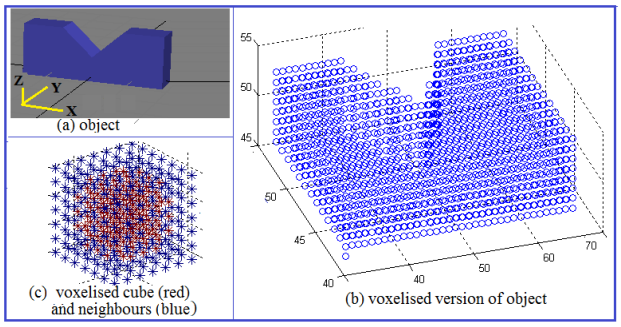
Fig. 11 Data associated with a voxel in 3D voxel array

4.1.1 Optimization in voxel conversion :

Optimization is done to reduce computation. Intersection of triangular plane (STL element) with grid vector is more precise if intersecting vector is not parallel or near parallel to surface. Generally grid vectors of any orientation i.e. either X, Y or Z can be used to intersect the STL triangle [25] but to ensure intersection by grid vector of appropriate orientation' in this case minimum bounding brick is formed with three nodes and minimum dimension orientation is found. Grid line in the same direction and coinciding with grid location in the plane orthogonal to probe vector direction are used as in fig. 12a and 12b to intersect the triangle for Boundary Voxel (BV) determination. Mark use of vectors of different direction in fig. 12a and 12b. Conventional approach would have used STL surface normal computation, computation of direction coefficient with X, Y and Z axes and comparison of the coefficient to find the nearest parallel direction. It would use 48 float operations (table1-7) whereas here only integer comparison is used to reduce computation. Further only grid vectors existing within face are used to avoid redundant vector –plane intersection test. While accessing the surface nodes , they are also tested for minimum x, maximum x, minimum y, maximum y and minimum z, maximum z of the entire object.



. Fig 12. Optimising intersection computation by minimising probe vectors.



. **Fig.13 Results of voxelisation of an object STL coded object.**
. **A cube object voxelised and neighbour formed around it (c).**

4.1.2 Voxel labelling :

In phase II the modified voxel array is analysed fully to label inner voxels. Voxels at constant Z (integer) levels in Z max > Z > Zmin zone have closed counters formed by boundary voxels of object (fig. 10c). The voxel data for a Z value is treated like 2D image as they form a slice of object. These have closed contours formed by earlier marked boundary voxels. By using raster scan method [6] the inner voxel are classified. Operation over Z min to Z max (computed earlier so only reused) results in full voxelisation of part. Object serial number is also appended as 'object index' with such labeled voxels. Gray voxels (boundary voxel) in fig 10c arise from side and those in fig. 10d from top surfaces of part in fig.10b. The blue voxels in 10c are inner voxels for the parts in the fig. 10b. The resulting array is the modified voxel array representing 'empty', boundary and 'inner' type voxels. Now the array is voxel based representation of workspace occupancy. Anothervoxelised model formation is shown in figure 13.

4.2 Neighborhood modeling algorithm

Subspace of cube shape around a single chosen object is formed from the 3D voxel array and referred as object voxel array OVA(x,y,z). Its size is based on the extent of desired neighborhood from the surface. All voxels belonging to other object falling within this subspace are reset to 'empty' type.

The floating analyzer zone 'FAZ' is formed as 8 connected neighboring voxel set around a center voxel FZ(i, j, k) . See figure 14. It has 2m+1 layers of (2m+1) x (2m+1) voxel grids. This 3D zone is indexed by indices i,j,k varying in range –m*i to +m*i , –m*j to +m*j and –m*k to +m*k along x,y and z directions respectively. Size of FAZ is determined by integer 'm, around the center voxel FZ(i=0, j=0, k=0). The value of FZ(i, j, k) may be 1or 0 for forming shaping element other than cube while maintaining high efficiency array based ordered processing. A '0' inhibits processing at the location.

The object voxel array OVA(x,y,z) is scanned. On finding a voxel with property as 'boundary', FZ is aligned with FZ(i=0, j=0, k=0) at OVA(x,y,z). A search within local neighbourhood

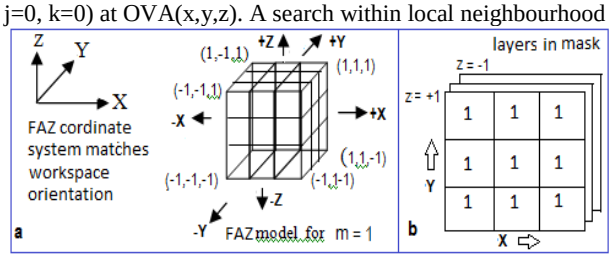


Figure 14, Floating analyser zone and mask values.

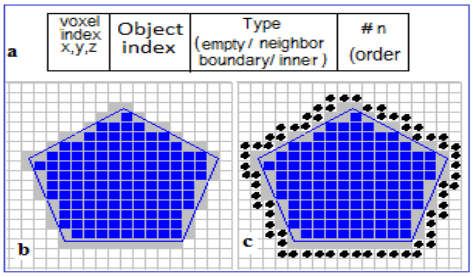


Figure 15 a. Data record associated with a voxel; c; neighbour processing results on object in b.

at $\mathbf{OVA_{(x,y,z)}}$ corresponding to **FZ** limits is carried out on voxel locations where condition $\mathbf{FZ_{(i,j,k)}}$ =1 is satisfied and $\mathbf{OVA_{(x+i,y+j,z+k)}}$ is a voxel with property 'empty'. The type field of such voxel is a converted to 'boundary' and a parameter value '1' is assigned. For example, when FZ has m=1 then all $\mathbf{OVA_{(x,y,z)}}$ voxels with 'empty' property existing within the (-1,-1,-1) to (+1,+1,+1) limits of i,j,k are changed if the mask $\mathbf{FZ_{(i,j,k)}}$ =1 condition is valid. (fig.14b). This process is continued over the entire x,y,z index space of $\mathbf{OVA_{(x,y,z)}}$.The resulting OVA with new 'neighbour' type in the type field and n=1 in 'order' field is shown in figure 15a. Colour coded 'inner'(blue) and 'boundary' (gray) voxel in one horizontal layer of original OVA appears in fig-15b. After above process, 'neighbour' order '1' (black) is also created (15c). This completes one level of boundary grading. For growing the graded neighbour zone, the process above is repeated in phase II on latest **OVA** by searching voxels in **OVA** modified array data (fig. 15a) that have property 'neighbour' and grade 'n=1' and converting all 'empty'voxels within **FZ** to 'neighbour' grade 2. Repeating this modification on **OVA** for 'n' times, creates voxelised object with 'n' graded neighbours surrounding it (fig. 16).

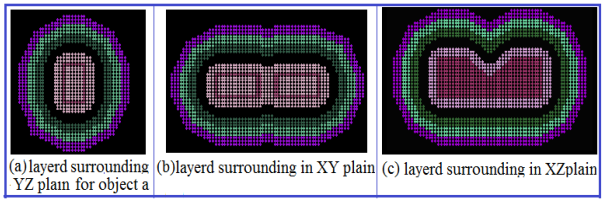


Fig.16 Results of 7 layer neighbour formation on object fig. 13a.

All the objects in workspace are converted to this form by forming their object voxel array OVAs and creating multi-graded neighbour coding in them. The number of OVA created so, will be equal to number of objects in workspace. These are mapped back into the 3D voxel array for the whole workspace. For improved throughput optimisation in computation is done easily (fig12). It may be noted that the originating object reference is inherited in voxel definition and so object specific mediation in its neighbourhood is feasible. One example is different treatment to approachable and unapproachable object types by mediator.

4.3 Benefits of the new approach

The graded neighbour forming process is done prior to tele-operation start and so time constraint is lenient. It is done from object definition CAD included in workspace definition so no search method is involved. For improved throughput, optimisation in computation is done easily (fig12). It may be noted that the originating object reference is inherited in voxel definition and so object specific mediation in its neighbourhood is feasible. One example is different treatment to approachable and unapproachable object types .

5. EXPERIMENTAL MEDIATION SETUP AND TEST REULTS

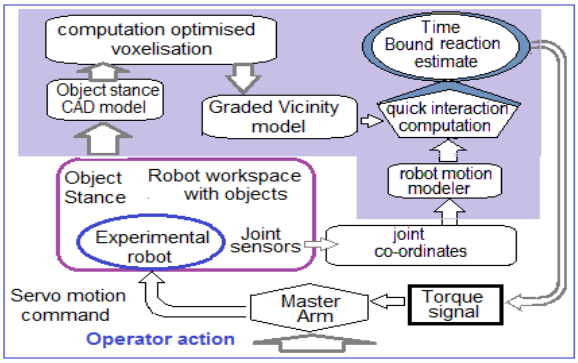


Figure 17. system with computation optimised modelling (shaded)

An experimental system (fig. 17) comprising voxel modeller working on principal referred above, a 5 axes Cartesian robot arm typically used in tel-manipulation in large cells (figure 19) , an electronic system capable of joint tracking, torque actuator based joystick and kinematics model based position estimator for robot arm tip was set up to test the approach. Voxelised version of workspace is created from CAD model (STL length parallel to X axis and base on horizontal table top. Various velocity profiles are fed to robot arm to approach the real version) instantiated by object position and orientation of actual object. In the experiment a part was placed with major specimen object. The joint positions of robot are tracked in real time by encoder and used by kinematics modeller to extract actual tool tip i.e. $\mathcal{R}$ location. By computation based on graded model explained in earlier section a force signal is computed and sent to the torque motors in joystick. The mediator computer system generates opposing action on joystick in part vicinity discouraging the operator to move further.

5.1 Quick acting mediation

In pre-run phase, desired model is formed by choosing n, d and l. Resulting impedances ρ_n are associated to the space around the part in voxelised workspace. In this arrangement, ρ_1 is the impedance closest to part surface and is assigned from the last segment of the cylinder and ρ_n are assigned from the respective segments with n as the number of bypasses in the model (fig. 7). In experiments here, n is 6.
The approach assigns spatial impedance to neighborhood which is then maintained by mediator processor module (fig.17) as property. In run time the probe tip position is computed from the kinematics model using the present joint parameters being followed by slave arm. This being the $\mathcal{R}$ position in workspace, corresponding ρ_n value is retrieved. $\mathcal{R}$ velocity 'ν' is computed from discrete time stamped positions at time regular intervals or from velocity sensors on slave robot arm. For same configuration master slave arms, master joints' sensor readings are also used instead. Applicable F is computed by very lean computation (7).

5.2 Performance of method in test set up

An experimental set-up was built using a tele-controlled set up that uses 5 axes robot [8] of Cartesian type (fig. 18,19). The tool tip can be oriented in space by rotations in two mutually orthogonal plane and located by 3 axis Cartesian positioning in a workspace of size1000x1000x600mm.

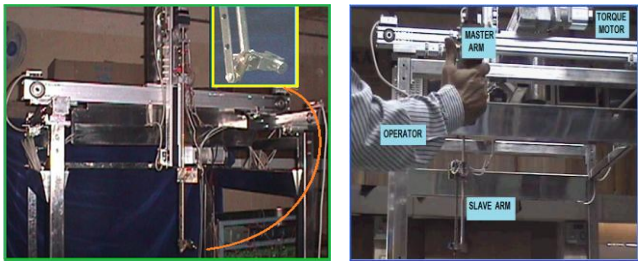


fig (18) Experimental 5 DOF Cartesian slave robot
Fig 19 (right), master

The tool tip can be oriented in space by rotations in two mutually orthogonal plane and located by 3 axis Cartesian positioning in a workspace of size1000x1000x600mm. The robot and associated control set-up is equipped with joint position and velocity sensing electronics. In 3D space of the robot 'Z' (height) is kept fixed at 300 mm. A solid cube placed in workspace coded with voxel approach appears as in fig.21 if a single horizontal plane with Z=300 is selected. The R motion is permitted in plane corresponding to this Z. For experiments, X motion is carried out at controlled speed along Y=20 coordinate. Model coding is done as described earlier. The parameter gap 'G' is mapped as 'k' (integer) numbers of voxel of length 'Vx'. This implies that k is scale factor between model and actual workspace voxel dimension expressed as an integer. Fluid resistance ρ forms a pattern and for k = 2, it appears as

$$\text{Array\_V}_{(x,20,30)} = \rho_1, \rho_1, \rho_2, \rho_2, \rho_3, \rho_3, \rho_4, \rho_4, \ldots\ldots\rho_n, \rho_n \ldots\ldots \quad (4)$$

The profile is depicted by different color code in fig. 20 .

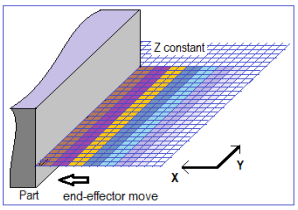


Fig.20, ρ based activation encoding of plane at Z=300 .

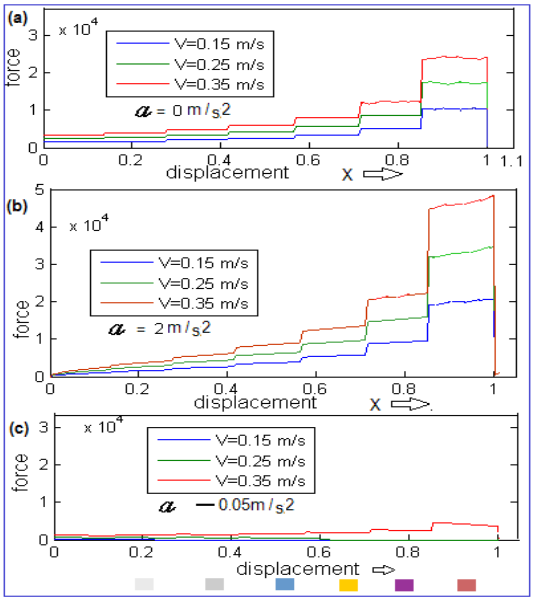


Fig.21 a; opposing force generated at different velocities for model coded as per equation (10) b; for accelerating motion c; for decelerating motion

Performance is assessed for models based on different natured gap '$\mathcal{G}$'. Equal gaps and varying gap result in valuable perfomance varieties. accelerations and constant deceleration are shown in fig. 21b and 21c. Color blocks below the horizontal axes (fig. 21,24) indicate correspondence to the same color coded voxels in activation schemes (fig. 20,22,23).

5.2.1 Effect of 'gap' variation between bypasses:

If value of gap 'G' is reduced then 'k' reduces and lowest value can be k =1 (fig.22). As effect of narrow gap, $\mathcal{R}$ finds frequent F change (fig 24a). Alternatively a combination of regular gaps away from object and narrow gap only near object results in more frequent occurrence of 'F' steps only very near the object. It provides enhanced indication of progressively nearer vicinity. The solution lies in designing 'G' of different sizes. Coding scheme with varying k (fig. 23) that has k=2 at head-end and k=1 near object gives desired result (fig, 24b).

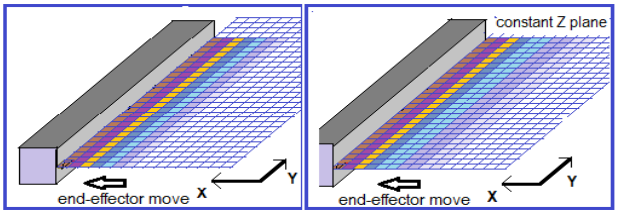


Fig. 22 encoding for narrow 'G' Fig. 23 unequal 'G' encoding

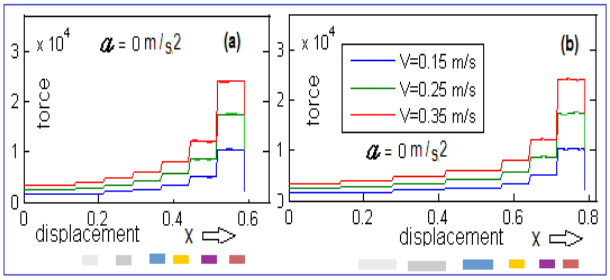


Fig. 24 Effect of changing '$\mathcal{G}$' a; regular gap with k=1 and b; with varying k, initially k=2 and later k=1

Figure (24.a) shows results for model using narrow uniform 'G' as compared to that in fig. (21a). Performance of non regular 'G' based (fig. 23) models is shown in fig 24b.

In experiments single axis move of master interface (fig. 19) was used. The force F was created by a 1.28 Nm low inertia motor operating in torque mode with geared force multiplication factor of 15 and toque dynamic range limited to 35% of full scale. Operator perceived the force effects distinctly in real time.

6. DISCUSSION ON RESULTS

Complete computation in run time for vicinity detection and corresponding effect creation was completed in time bound manner, within T=2ms.

This is achieved by relegating complex computations to offline workspace model generation stage. Real time computation is minimal and is as described in fig. 25. Instead of estimating vicinity by computing spatial relationship to complex part, now vicinity dependant pre-computed ρ data at location of $\mathcal{R}$ is sensed at run time. The accessing of this data is made time efficient by using structured array based assignment [26] and quick index determination in online control loop. Float operations are limited to forward kinematics solving [27] for given joint parameters and computing equation (3).

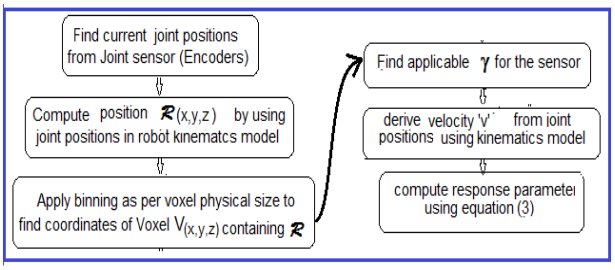


Fig. 25, The vicinity based feedback is computed on line inside control loop with very lean computation.

For voxel formation , though the process is off line, care has been taken to achieve efficient computing by making the process object centric. Instead of exhaustive gridline intersection search with all possible STL surfaces, the optimized process takes up surfaces of a single object at a time and processes them in order of their occurrence in CAD model, finds optimum grid orientation for intersection and filters in positively intersecting grid lines by 2D geometry analysis (36 float operations) rather than 3D method (68 float operations) to save computation. Similarly instead of orienting the voxel model of object to the object stance, object model itself that has relatively lesser nodes is translated and rotated to align model pose to object's real pose in workspace.

A notable aspect of the method is that run time performance is not limited by number of objects in workspace. Fine grain response can be generated by reducing voxel size. Finer grid increases trueness of the model but grid spacing must be controlled based on objective to reduce data set size particularly in real time application to avoid memory shuffling for getting pertinent data in RAM. Reducing voxel dimension to half leads to 8 times voxel data for the same work space .

The forward kinematics computation in the present setup is minimal owing to the robot configuration. In applications with 7 degree freedom articulated arms too the forward kinematics computation load is limited to under 160 float operations in optimized form rather than metrics solving that needs over 598 float operations. All computation is managed in bounded time.

7. CONCLUSION

The efficacy of the method devised here, lies in its capability to support variety of vicinity response creation by the mediator system while incurring very low run time computation burden. The mediation created by the method can even be object specific.

An interesting factor is the need for dynamic change of 'δ ' i.e. the vicinity extent with speed of $\mathcal{R}$. At higher speed $\mathcal{R}$ must have enough time to react to the vicinity without hitting the object surface in spite of its increased momentum. This is achieved by forming more graded layers extending to outer side of close vicinity at neighbor forming stage and taking recourse to early effect initiation with wider ρ profile.

For further improvement of response at run time, investigation on formulation of temporal or spatial filter is underway. Care is necessary to minimize runtime computation load and optimizing memory size for primary RAM resident workspace voxel model of larger sized robot workspaces

8. ACKNOWLEDGEMENTS

Author thankfully acknowledges the support extended by Sri G. Baluni for the experimental work . Dr . L.M. Gantayet and Dr. B.K.Datta have encouraged the work.

9. REFERENCES

[1] M. C. Cavusoglu, A. Sherman, and F. Tendick. "Design of bilateral teleoperation controllers for haptic exploration and telemanipulation of soft environments". IEEE Transactions on Robotics and Automation, 18(4):641–647, 2002.

[2] H. Kazerooni, "A Controller Design Framework for Telerobotic Systems", IEEE Transactions on Control Systems Technology, March 1993, pp. 50-62

[3] D. A. Lawrence, "Stability and Transparency in Bilateral Teleoperation" IEEE Transactions - Robotics and Automation, 9(5):624-637, 1993.

[4] Speich, J. E., Fite, K., and Goldfarb, M. "Method for Simultaneously Increasing Transparency and Stability Robustness in Bilateral Telemanipulation". Proceedings of IEEE International Conference on Robotics and Automation, pp. 2671-2676, April 2000.

[5] Tetsuo Kotoku "A Predictive Display and its Application to Remote Manipulation System with Transmission Time Delay" Proceedings - IEEE/RSJ International Conference on Intelligent Robots and Systems Raleigh, NC July 7-10, 1992 pg239-246

[6] Computer Graphics PRINCIPLES AND PRACTICE Second Edition James D. Foley Andries van Dam, Steven K. Feiner John F. Hughes Chapter 3 Pg.72-92

[7] Cheuk Yiu and Satyandra K. Gupta. Retrieving matching CAD models by using Partial 3D point clouds", Computer Aided Design & Applications, Vol-4, No-5, 2007, pp 629-838.

[8] C. K. Chua; Leong, K. F.; Lim, C. S. "Rapid Prototyping: Principles and Applications" World Scientific Publishing Co, ISBN 981-238-117-1 Chapter 6 Page 237

[9] J.D. Foley, A. van Dam, S.K. Feiner, J.F. Hughes, Computer Graphics: Principles and Practice (2nd Ed.), Addison–Wesley 1990 ISBN-10: 0201848406

[10] E.G. Gilbert, D.W. Johnson, and S. Keerthi, "A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space", IEEEJ. Robot. Automat. 4(2): 19S-203; April 1988.

[11] J. O'Rourke and N. Badler, "Decomposition of Three-Dimensional Objects into Spheres", IEEE. Trans. Pattern Anal. Mach. Intell. PAMI-1(3): 295-305; 1979.

[12] J. Tomero, J. Hamlin, and R.B.Kelly, "Spherical-Object Representation and Fast Distance Computation for Robotic Applications", Proceedings of the 1991 IEEE International Conference on Robotics and Automarion, April 1991. pp. 1602-1608.

[14] Shahram Payandeh , Zoran Stanisic, "On Application of Virtual Fixtures as an Aid for Telemanipulation and Training", Proceedings of the 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, p.18, March 24-25, 2002

[15] Shinsuk Park , Robert D. Howe , David F. Torchiana, "Virtual Fixtures for Robotic Cardiac Surgery", Proceedings of the 4th International Conference on Medical Image Computing and Computer Assisted Intervention, pp1419-1420, October 14-17, 2001

[16] J. K. Mukherjee et al, "An Observer Based Safety Implementation in Tele-Robotics" Proceedings of International Conference TIMA-2011, Jan 2011, Chennai India

[17] J. K. Mukherjee, "A Bounded Time Impending Hit Detection Method for Tele-manipulator" Intl. conf. ICCA-2012 Jan27-31-Pondichery, India

[18] J.K. Mukherjee, "AI Based Tele-Operation Support Through Voxel Based Workspace Modeling and Automated 3D Robot Path Determination", Conference on Convergent Technologies for the Asia-Pacific Region, vol. 4 of 4, Conf. 18, Oct. 15-17, 2003, pp.305-309

[19] J. K. Mukherjee, "Virtual Sensor Model for Proximity Perception" Proc of International Conference ICST 2012 Dec 18-21, 2012, Kolkata India (in-press)

[20] Kaufman, A., Cohen, D. and Yagel, R., "Volume Graphics", IEEE Computer, Vol. 26, No. 7, pp. 51-64, July 1993.

[21] Kaufman, A., Shimony, E., "3D Scan-Conversion Algorithms for Voxel Based Graphics", Proceedings of the 1986 workshop on Interactive 3D graphics, pp. 45-75, 1987.

[22] Cohen Or, D., Kaufman, A., "Fundamentals of Surface Voxelization", Graphical Models and Image Processing, 57, 6 (November 1995), 453-461.

[23] Arie Kaufman, Eyal Shimony, "3D scan-conversion algorithms for voxel-based graphics", Proceedings of the 1986 workshop on Interactive 3D graphics, p.45-75, January 1987, Chapel Hill, North Carolina, United States

[24] D. Cohen-Or and A. Kaufman. "3D line Voxelization and Connectivity control". IEEE Computer Graphics and Applications, 17(6):80—87, 1997.

[25] Huang, J., Yagel, R., Filippov, V., Kurzion, Y., "An Accurate Method for Voxelizing Polygon Meshes", Proceedings of the 1998 IEEE Symposium on Volume Visualization, pp. 119-126, 1998.

[26] Samel H. "Design and analysis of spatial data structures in Computer graphics, Image processing and GIS", Addison Wesley, Reading M. A. 1990